**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ,
МОЛОДЕЖИ И СПОРТА УКРАИНЫ**

**ХАРЬКОВСКАЯ НАЦИОНАЛЬНАЯ АКАДЕМИЯ
ГОРОДСКОГО ХОЗЯЙСТВА**

Б. И. Погребняк

КОМПЬЮТЕРНАЯ ТЕХНИКА И ПРОГРАММИРОВАНИЕ

КОНСПЕКТ ЛЕКЦИЙ

(для студентов 1-го курса заочной формы обучения образовательно-квалификационного уровня бакалавр, направления подготовки 6.070101 «Транспортные технологии» (по видам транспорта))

**ХАРЬКОВ
ХНАГХ
2011**

Погребняк Б. И. Компьютерная техника и программирование: конспект лекций (для студентов 1-го курса заочной формы обучения образовательно-квалификационного уровня бакалавр, направления подготовки 6.070101 «Транспортные технологии» (по видам транспорта)) / Б. И. Погребняк; Харьк. нац. акад. гор. хоз-ва. – Х.: ХНАГХ, 2011. – 173 с.

Автор: к.т.н., доц. Б. И. Погребняк

Рецензент: доц., к. ф.-м. н. А. Б. Костенко

Конспект лекций построен в соответствии с требованиями кредитно-модульной системы организации учебного процесса и согласован с ориентировочной структурой содержания учебной дисциплины, рекомендованной Европейской Кредитно-Трансферной Системой (ECTS).

Рекомендовано для студентов специальностей «Транспортные технологии».

Утверждено на заседании кафедры Прикладной математики и информационных технологий, протокол № 1 от 30 августа 2010 г.

ОГЛАВЛЕНИЕ

ЛЕКЦИЯ №1

ХАРАКТЕРИСТИКА И КЛАССИФИКАЦИЯ СРЕДСТВ КОМПЬЮТЕРНОЙ ТЕХНИКИ

План

1. Эволюция информационных систем
2. Понятие информации
3. Способы получения информации
4. Свойства информации
5. Передача информации
6. Структурная схема компьютера
7. Физическая организация хранения информации на магнитных дисках
8. Уровни памяти компьютера

1. Эволюция информационных систем

Дисциплина Компьютерная техника и программирование тесно переплетается с информатикой – наукой о способах и методах представления, обработки, передачи и хранения информации с помощью компьютера. Сам термин «информатика» происходит от французского Informatiqe. Впервые он возник в 60-е годы XX столетия путем слияния двух слов Informacion (Информация) и Automatique (Автоматика) для обозначения деятельности по автоматизированной обработке информации с помощью ЭВМ. Кроме Франции термин информатика широкое распространение получил также в большинстве стран Европы. В то же время в англоязычных странах аналогичный круг проблем обозначается термином «Computer Science», что означает буквально «Компьютерная наука».

Международный конгресс по информатике 1978 г. предложил следующее определение: «Понятие информатики охватывает области, связанные с разработкой, созданием, использованием и материально-техническим обслуживанием систем обработки информации, включая машины, оборудование, математическое обеспечение, организационные аспекты, а также

комплекс промышленного, коммерческого, административного и социального воздействия». В бывшем СССР термин информатика утвердился в 1983 г., когда в Академии наук было создано отделение «Информатики, вычислительной техники и автоматизации». Тогда же информатика стала трактоваться как «комплексная научная и инженерная дисциплина, изучающая все аспекты разработки, проектирования, создания, оценки, функционирования основанных на ЭВМ систем переработки информации, их применения и воздействия на различные области социальной практики».

Информатика как наука сложилась сравнительно недавно. Ее развитие самым тесным образом связано с появлением в середине XX века электронных вычислительных машин, которые являются универсальным средством для хранения, обработки и передачи информации. В качестве источников информатики обычно называют две науки – документалистику и кибернетику. Основным предметом документалистики является изучение рациональных средств и методов повышения эффективности документооборота. Она сформировалась в конце XIX века в связи с бурным развитием производственных отношений, а расцвет ее пришелся на 30-е годы XX века.

Кибернетика, как наука об общих закономерностях управления и связи в различных системах – технических, социальных, биологических, возникла и начала бурно развиваться после второй мировой войны. Создателем кибернетики принято считать американского ученого Норберта Винера, опубликовавшего в 1948 г. свою знаменитую книгу «Кибернетика, или управление и связь в животном и машине». В ней были показаны пути создания общей теории управления и заложены основы методов рассмотрения проблем управления и связи для различных систем с единой точки зрения. Развиваясь одновременно с развитием электронно-вычислительных машин, кибернетика со временем превращалась в более общую науку о преобразовании информации. Сам же термин «кибернетика» происходит от греческого слова kyberneticos – искусный в управлении.

Обработка информации при помощи компьютера осуществляется аналогично получению фарша в обыкновенной мясорубки. Если на вход мясорубки кладут ломтики мяса, а на выходе получают то же мясо, только перекрученное – фарш, то на вход компьютера поступает исходная информация, а на выходе получается уже обработанная – выходная информация. Тогда схема обработки информации на компьютере в самом общем виде может быть представлена следующим образом (рис. 1.1).

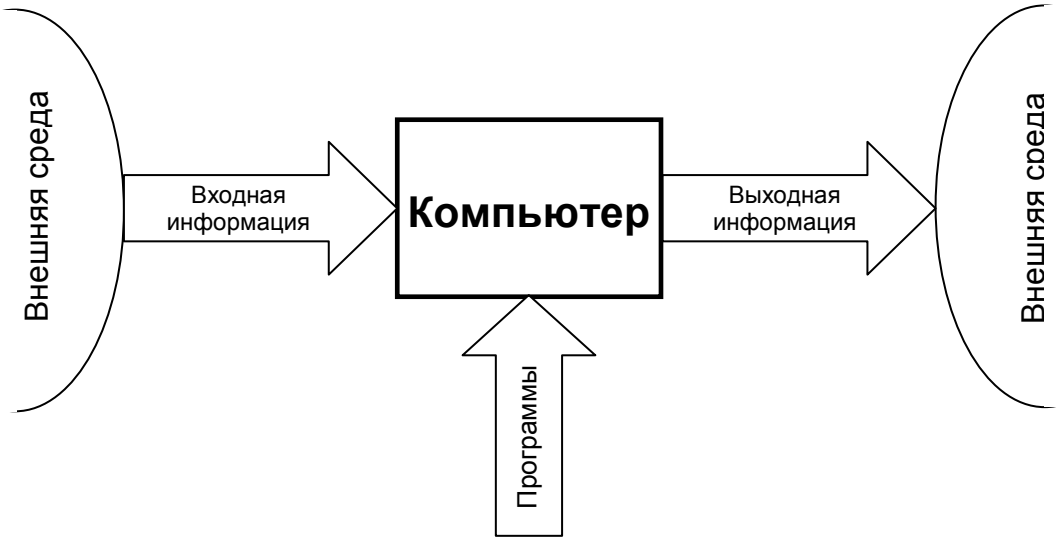


Рис. 1.1 – Общая схема обработки информации на компьютере

Из рис. 1.1 видно, что в этом процессе участвуют три составляющие:

1 информация,
2 компьютер и
3 программы.

2. Понятие информации

Термин «информация» происходит от латинского слова «informatio», что означает сведения, разъяснения, изложение, осведомленность и т.д. Само понятие информация является одним из фундаментальных в современной науке вообще, и базовым для информатики в частности. Наряду с такими понятиями, как вещество, энергия, пространство и время, оно рассматривается в качестве важнейшей сущности мира, в котором мы живем. Его нельзя определить через

более простые понятия. В математике, например, аналогичными «неопределяемыми» понятиями являются «точка» или «прямая», относительно которых можно сделать некоторые утверждения, но сами они не могут быть определены с помощью более элементарных понятий. В слово «информация» вкладывается различный смысл в экономике, науке, технике, различных житейских ситуациях.

В простейшем бытовом понимании (житейский аспект) под информацией понимают сведения об окружающем нас мире и протекающих в нем процессах, воспринимаемые человеком или специальными устройствами. Под информацией в технике понимают сообщения, передаваемые в форме знаков или сигналов. Под информацией в семантическом (смысловом) аспекте понимают сведения, обладающие некоторой новизной. Под информацией в кибернетике, по определению Норберта Винера понимают ту часть знаний, которая используется для ориентирования, активного действия, управления, т.е. в целях сохранения, совершенствования, развития системы. В середине XX века слово «информация» в узком техническом смысле ввел Клод Шеннон применительно к теории связи и передачи сигналов, которая получила название «Теория информации». Под информацией в ней понимают не любые сведения, а лишь те которые, снимают полностью или уменьшают существующую неопределенность, т.е. информация связана со снятием неопределенности.

Применительно к компьютерной обработке под информацией понимают некоторую последовательность кодов, несущую смысловую нагрузку и представленную в понятном компьютеру виде.

3. Способы получения информации

Информация о любом событии, объекте, его параметрах, и т.д. может быть получена одним из следующих способов:

1. *Опыт*. Является важнейшим методом получения информации. Он состоит в том, что ежедневно каждый человек, вольно, или невольно, накапливает определенное количество информации – некоторый опыт. В прошлом этот метод был основным и единственным для получения информации в жизни и

развитии человека. Множество замечательных достижений было получено опытным путем, в процессе накопления опыта и выводом определенных умозаключений. Например, какого совершенства достигли древние мастера Китая в изготовлении фарфора, тульские оружейники – в изготовлении оружия.

2. *Эвристический подход*. Его название происходит от слова «эврика», что в переводе с древнегреческого означает «Я нашел!». При эвристическом подходе проводят многократные эксперименты, после которых отбирают наиболее удачные варианты. Поэтому он еще называется «Методом проб и ошибок». Однако этот метод довольно длительный и трудоемкий, а потому недостаточно эффективный.
3. *Целенаправленный поиск*. Он характеризуется тем, что производится не беспорядочный перебор всех возможных вариантов, а анализируются известные достижения в конкретной области, планируются и проводятся опыты. В результате применения целенаправленного поиска человечество создало новые материалы и процессы, ранее не известные в природе. Этому способу получения информации способствует развитие современной техники, которая позволяет обрабатывать огромные объемы информации и получать при этом все новые и новые результаты.

Информация может быть получена так же:

1. путем наблюдения за объектом,
2. вычислительного эксперимента над ним
3. или путем логического вывода.

В связи с этим информация делится на доопытную (или априорную), и послеопытную, (или апостериорную), полученную в результате проведенного эксперимента.

4. Свойства информации

Каждая научная дисциплина рассматривает те свойства информации, которые ей наиболее важны. С точки зрения компьютерной обработки наиболее важными представляются следующие свойства:

1. *Объективность*. Информация объективна, если она не зависит от чьего – либо мнения.
2. *Достоверность*. Информация достоверна, если она отражает истинное положение дел.
3. *Полнота*. Информацию можно считать полной, если ее достаточно для понимания и принятия решения.
4. *Актуальность* – важность, существенность для настоящего времени.
5. *Адекватность* – определенный уровень соответствия создаваемого с помощью полученной информации образа реальному объекту, процессу, явлению.

5. Передача информации

Основной особенностью информации является то, что она является категорией не материальной. Следовательно, для существования и распространения в материальном мире информация должна быть обязательно связана с какой-либо материальной основой – без нее информация не может проявиться, передаваться и сохраняться. Поэтому материальный объект или среду, которые служат для представления или передачи информации, называют *материальным носителем*. Материальным носителем информации может быть бумага, воздух, лазерный диск, электромагнитное поле и пр. Изменение характеристики материального носителя, которое используется для представления информации, называют *сигналом*, а значение этой характеристики, отнесенное к некоторой шкале измерений, называется *параметром сигнала*. Например, при разговоре материальным носителем является воздух, сигналом – звуковая волна, которая в нем распространяется, а параметрами сигнала – высота и громкость звука.

Однако одиночный сигнал не может содержать много информации. Поэтому для передачи информации используется ряд следующих друг за другом сигналов. Такая последовательность сигналов называется *сообщением*. Примерами сообщений являются музыкальное произведение, телепередача, команды регулировщика на перекрестке, текст, распечатанный на принтере,

данные, полученные в результате работы программы и т.д. Можно сказать, что сообщение выступает в качестве материальной основы для представления информации при передаче. Следовательно, сообщение служит переносчиком информации, а информация является содержанием сообщения.

Сообщения, передающие одну и ту же информацию, образуют класс эквивалентных сообщений. Например, прогноз погоды может быть получен по радио, из газеты, по телевидению из Интернета и т.д. В то же время одно и то же сообщение может передавать совершенно различную информацию. Примером может служить передача в 1936 г. по радио фразы «Над всей Испанией безоблачное небо», которое для непосвященных людей имело смысл прогноза погоды, а для других – сигналом к началу военных действий.

Соответствие между сообщением и содержащейся в нем информацией называется *правилом интерпретации сообщения*. Таким образом, одно и то же сообщение, по-разному интерпретированное, может передавать разную информацию. Правило интерпретации может быть известно лишь ограниченному кругу лиц. Связь между сообщением и информацией особенно отчетлива в криптографии: никто посторонний не должен суметь извлечь информацию из передаваемого сообщения.

Информация не может существовать без наличия источника и приемника (получателя, потребителя) информации. *Источник информации* – это субъект или объект, порождающий информацию и представляющий ее в виде сообщения. *Приемник информации* – это субъект или объект, принимающий сообщение и способный правильно его интерпретировать.

В этих определениях сочетание «субъект» или «объект» означает, что источники и приемники информации могут быть одушевленными (человек, животные) или неодушевленными (технические устройства, природные явления). Для того чтобы объект (или субъект) считался источником информации, он должен не только ее породить, но и иметь возможность создать сообщение. Например, если человек что-то придумал, но держит это в своей голове, он не является источником информации. Однако он им становится, как

только свою идею изложит на бумаге (в виде текста, рисунка, схемы и пр.) или выскажет словами.

В определении приемника информации важным представляется то, что факт приема сообщения еще не означает получение информации. Информация может считаться полученной только в том случае, если приемнику известно правило интерпретации сообщения. Другими словами, понятия «приемник сообщения» и «приемник информации» не тождественны. Например, слыша речь на незнакомом языке, человек оказывается приемником сообщения, но не приемником информации.

Конечным источником и потребителем информации при ее обмене является человек. Следовательно, воспринимать сообщения мы можем только посредством одного из пяти органов чувств, или некоторой их группой. Это не означает, однако, что человек не может использовать для приема и передачи информации какие-то иные сигналы, непосредственно им не воспринимаемые, например радиоволны. В этом случае человек-источник использует промежуточное устройство, преобразующее его сообщение в радиоволны – радиопередатчик, а человек-приемник – другое промежуточное устройство – радиоприемник, преобразующий радиоволны в звук. Такой подход заметным образом расширяет возможности человека в осуществлении передачи и приема информации. Промежуточные устройства-преобразователи получили название *технические средства связи*, а в совокупности с соединяющей их средой они называются *каналом связи* (*каналом информации*, *информационным каналом*). К ним относятся телеграф, телефон, радио и телевидение, компьютерные телекоммуникации и пр. Передача информации по каналам связи часто сопровождается воздействием помех, вызывающих искажение и потерю информации (рис 1.2.).

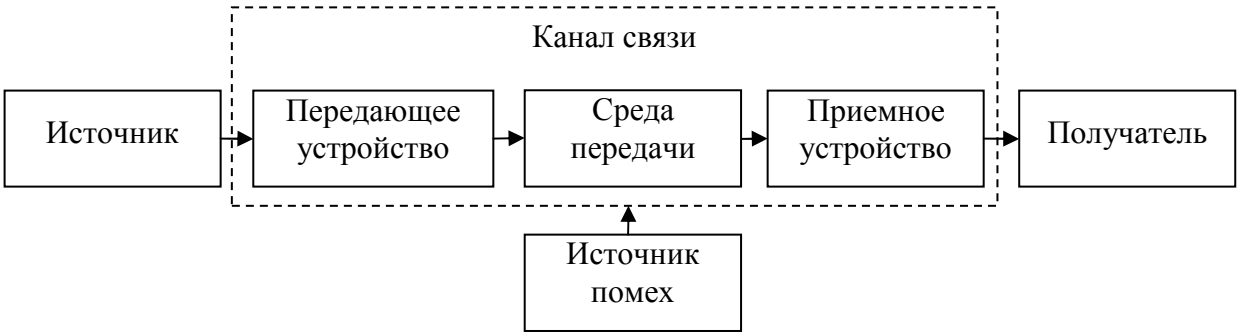


Рис. 1.2 – Общая схема передачи информации

Примером канала связи может служить почта. Информация, записанная в виде текста на листе бумаги, помещается в конверт, опускается в почтовый ящик, затем извлекается из него и перевозится в почтовое отделение, где аналогичные отправления сортируются по направлениям назначения. Далее это отправление при помощи различных видов транспорта (автомобилей, поездов, самолетов и т.д.) доставляется в пункт назначения, где аналогичная корреспонденция распределяется по почтовым отделениям, откуда непосредственно доставляются адресату. Таким образом, почтовый канал связи включает в себя: конверт, транспорт, сортировочные машины, и почтовых работников. Информация, переданная при помощи этого канала связи, не искажается.

Другим примером может служить телефон. При этом источником сигнала является говорящий. Кодирующим устройством, преобразующим слова говорящего в электрические импульсы, является микрофон. Канал, по которому передается информация – телефонные провода. Та часть трубки, которую мы подносим к уху, является декодирующим устройством, в котором электрические сигналы снова преобразуются в звук. Таким образом информация поступает в «принимающее устройство» – ухо человека на другом конце провода. Канал связи включает в себя телефонные аппараты, соединительные провода и коммутирующие устройства АТС. Особенностью этого информационного канала является то обстоятельство, что при поступлении в него информации в виде звуковых волн (речи), он преобразуется

в электрические колебания, и лишь затем – передается. Такой канал называется *каналом с преобразованием информации*.

Еще один пример канала связи – компьютер. Его так же можно рассматривать как информационный канал с преобразование информации, поскольку, информация, поступающая с внешних устройств (клавиатура, микрофон и т.д.) преобразуется во внутренний код, обрабатывается, а затем снова преобразуется к виду, который может воспринимать устройство вывода (монитор, принтер и т.д.) и передается на них.

По информационным каналам связи могут передаваться два типа сигналов:

1. непрерывные (рис 1.3а) и
2. дискретные (рис. 1.3б).

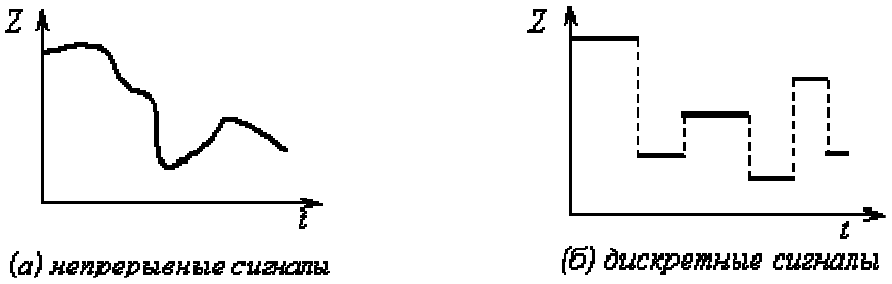


Рис. 1.3 – Непрерывные и дискретные сигналы

Сигнал называется *непрерывным* (или аналоговым), если его параметр может принимать любое значение в пределах некоторого интервала. Примерами непрерывных сигналов являются речь, музыка, изображение, показание термометра (параметр сигнала – высота столба спирта или ртути – имеет непрерывный ряд значений) и пр.

Сигнал называется *дискретным* (или цифровым), если его параметр может принимать конечное число значений (при этом все они могут быть пронумерованы) в пределах некоторого интервала. Сообщение, передаваемое с помощью таких сигналов, то же называется дискретным, и информация, передаваемая источником в этом случае, также является дискретной. Пример дискретного сообщения – процесс чтения книги, информация в которой

представлена текстом, т.е. дискретной последовательностью отдельных знаков (букв).

Принципиальным и важнейшим различием непрерывных и дискретных сигналов является то, что дискретные сигналы можно обозначить, т.е. приписать каждому из конечного числа возможных значений сигнала *знак*, который будет отличать данный сигнал от другого. *Знаком* называется элемент некоторого конечного множества отличных друг от друга сущностей. Природа знака может любой – жест, рисунок, буква, сигнал светофора, определенный звук и т.д. Его природа определяется носителем сообщения и формой представления информации в сообщении.

Вся совокупность знаков, используемых для представления дискретной информации, называется *набором знаков* – т.е., набор есть дискретное множество знаков. Набор знаков, в котором установлен порядок их следования, называется *алфавитом*. То есть, *алфавит – это упорядоченная совокупность знаков*. Порядок следования знаков в алфавите называется *лексикографическим*. Благодаря этому порядку между знаками алфавита можно установить отношения «больше–меньше». Для двух знаков «а» и «я» принимается, что ***а*** < ***я***, если порядковый номер у ***а*** в алфавите меньше, чем у ***я***. Примером алфавита так же может служить совокупность арабских цифр 0,1…9. С помощью такого алфавита можно записать любое целое число в системах счисления от двоичной до десятичной. Поскольку при передаче сообщения параметр сигнала должен меняться, очевидно, что минимальное количество различных его значений равно двум и, следовательно, *алфавит может содержать минимум два знака*. Такой алфавит называется *двоичным*.

 Представляется важным еще раз подчеркнуть, что *понятия знака и алфавита можно отнести только к дискретным сообщениям*.

Любое непрерывное сообщение может быть представлено непрерывной функцией, заданной на некотором интервале. Непрерывное сообщение можно преобразовать в дискретное. Эта процедура называется *дискретизацией*. Из бесконечного множества значений параметра непрерывного сигнала

выбирается их определенное число, которое приближенно может характеризовать остальные значения. Для этого диапазон изменения функции разбивается на отрезки равной длины и на каждом из этих отрезков значение функции принимается постоянным и равным (например, среднему значению на каждом отрезке). В итоге получается конечное множество чисел. Таким образом, любое непрерывное сообщение может быть представлено как дискретное, иначе говоря, последовательностью знаков некоторого алфавита.

Возможность дискретизации непрерывного сигнала с любой желаемой точностью (для возрастания точности достаточно уменьшить шаг) принципиально важна с точки зрения информатики. Компьютер – цифровая машина, т.е. внутреннее представление информации в нем дискретно. Поэтому дискретизация входной непрерывной информации позволяет сделать ее пригодной для компьютерной обработки.

Существуют, однако, и другие вычислительные машины – аналоговые ЭВМ. Они используются обычно для решения задач специального характера и столь широкого распространения, как цифровые, не получили. Эти ЭВМ в принципе не нуждаются в дискретизации входной информации, так как ее внутреннее представление у них непрерывно. В этом случае все наоборот – если внешняя информация дискретна, то ее «перед употреблением» необходимо преобразовать в непрерывную.

6. Структурная схема компьютера

Любой компьютер «стоит на трех китах». Это:

1. центральный процессор,
2. оперативная память и
3. внешние устройства.

Более подробно структурная (логическая) схема современного компьютера (аппаратного или, технического обеспечения, или электронной цифровой вычислительной машины – ЭЦВМ) представлена на рис. 1.4.

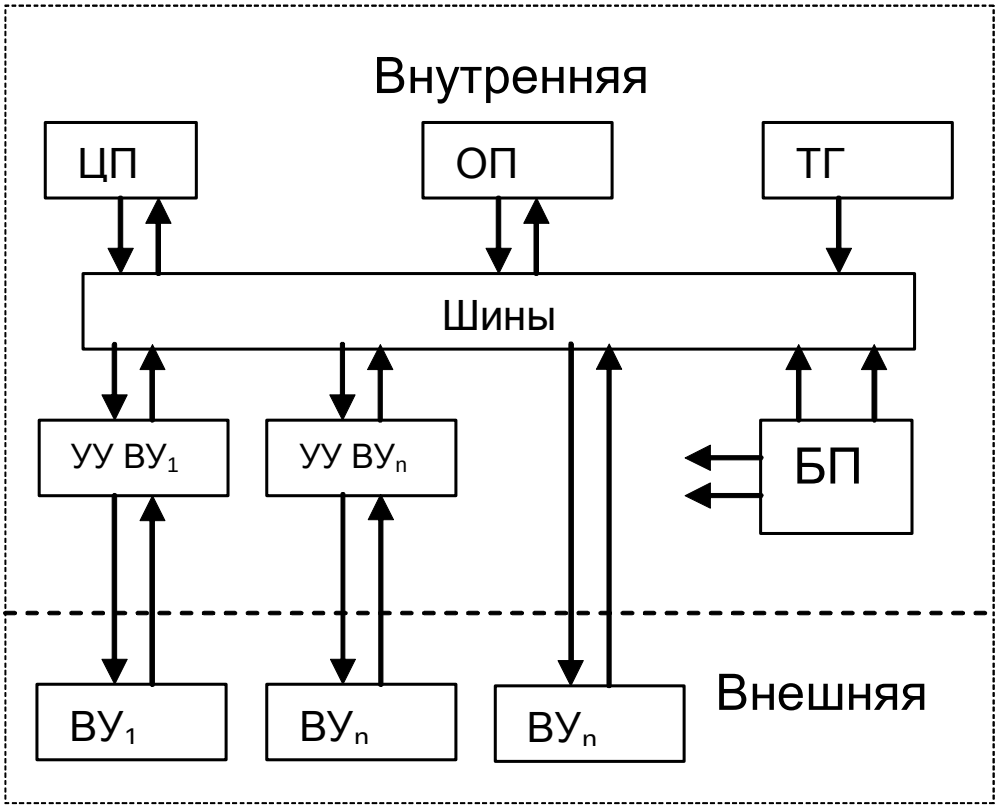


Рис. 1.4 – Структурная схема компьютера

Он состоит из двух классов устройств:

1. *Внутренних* (*центральных*), которые обеспечивают весь процесс обработки информации, и
2. *Внешних* (*периферийных*), предназначенных для передачи информации из внешней среды в компьютер и обратно, а также для длительного ее хранения.

Основным устройством компьютера является *центральный процессор* (ЦП). Он определяет все остальные его характеристики. Именно в нем происходит обработка информации. Совокупность всех возможных операций, которые процессор может выполнять над данными, образуют так называемую *систему (набор) команд* процессора.

Процессоры, имеющие одинаковые системы команд, так же *совместимы и на программном уровне*. Это значит, что программа, написанная для одного процессора, будет выполняться и на всех остальных процессорах с такой же системой команд. Процессоры, имеющие различные системы команд, как правило, полностью, или частично, несовместимы на программном уровне.

Некоторые процессоры, имеющие ограниченную совместимость, образуют так называемое *семейство* (*клан*) *процессоров*. Так, например, многие процессоры корпорации Intel относятся к так называемому семейству x86. Родоначальником этого семейства был 16-разрядный Intel 8086, на базе которого были собраны первые персональные компьютеры – знаменитые IBM PC. Впоследствии выпускались процессоры Intel 80286, Intel 80386, Intel 80486, Intel Pentium. Семейство процессоров Intel Pentium так же имеет несколько модификаций: Intel Pentium Pro, Intel Pentium II, Intel Celeron, Intel Xeon, Intel Pentium III, Intel Pentium 4 и другие. Все они, а также многие модели процессоров корпораций AMD и Cyrix, относятся к семейству x86 и обладают программной совместимостью по принципу «снизу вверх».

Принцип совместимости «снизу вверх» – это пример удачной совместимости, когда каждый последующий процессор «понимает» все команды своих предшественников, но не наоборот. Благодаря такой совместимости на современном компьютере можно выполнять программы, созданные ранее для любого из предшествующих компьютеров, принадлежащего к той же аппаратной платформе.

Примером полной несовместимости по системе команд с процессорами семейства x86 являются процессоры корпорации Motorola. На их основе компанией Apple создано целое семейство компьютеров Macintosh. Хотя по назначению и классу решаемых задач они точно такие же, как и IBM PC, но программы, созданные для IBM PC на компьютерах Macintosh работать не будут, как, впрочем, и наоборот.

Программная совместимость процессоров является одной из важнейших их характеристик. На рис. 1.5 представлен график изменения стоимости производства программ и компьютеров во всем мире с течением времени.

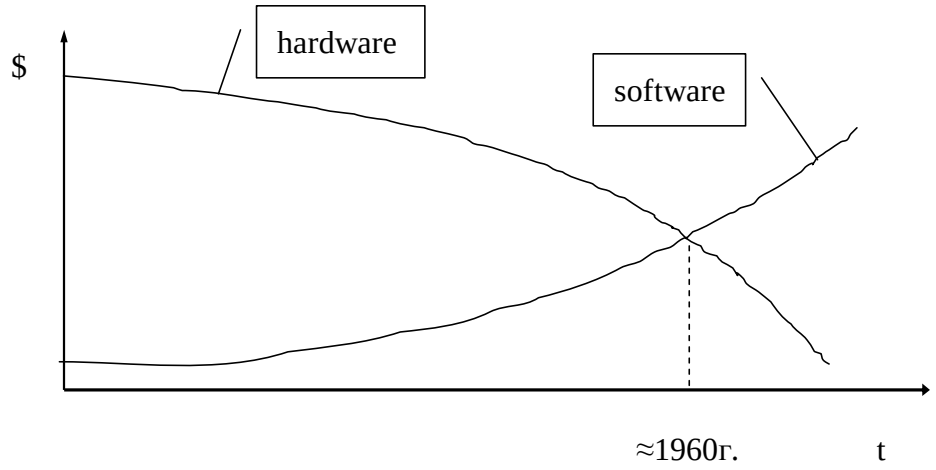


Рис. 1.5 – Тенденция изменения стоимости программных и аппаратных средств компьютера

Он демонстрирует то, что с течением времени затраты на создание программных средств увеличиваются, а компьютеров – уменьшаются. Из этого следует, что в подавляющем большинстве случаев, если некоторая программа перестала удовлетворять по быстродействию, гораздо дешевле заменить компьютер на более совершенный, нежели переписывать существующую программу. Соответственно, при замене одного компьютера на другой, с отличной от первого системой команд, придется нести и дополнительные (многократно превышающие стоимость самого компьютера) затраты по созданию для него необходимых программ.

Поскольку работой компьютера целиком и полностью управляют программы, то необходимо обеспечить легкий доступ процессора к последним. Тогда возникает вопрос: где лучше всего хранить программу? Американскому математику венгерского происхождения *Джону фон Нейману* принадлежит замечательная идея (теперь она кажется тривиальной): поместить программу вместе с данными, которые она обрабатывает, в *оперативную память* компьютера. Таким образом, одна и та же память используется для хранения и объектов, и «рецептов» вычислительных процессов, т.е. и данных, и программ.

Очевидно, что из самой концепции – хранить программу в оперативной памяти – следует, что команды также должны кодироваться. Каждая команда представляется кодом операции («сложить», «вычесть», «сравнить» и т.д.) и в

некоторых случаях – операндами. Если операнды представляют *адреса ячеек* оперативной памяти, и если предположить, что адреса ячеек – это целые числа 0, 1, 2, 3 и т.д., то проблема кодирования программы по существу решена. Каждая программа может быть представлена последовательностью чисел (или групп чисел) и, следовательно, может храниться в оперативной памяти компьютера.

Тогда, исходя из общей идеи совместного хранения в оперативной памяти и программ и данных, управление компьютером (и, следовательно, обработка информации на нем) выполняется самым тривиальным образом – процессор последовательно одна за другой извлекает команды из оперативной памяти и подчиняется им. Говорят, что процессор *выполняет* команды. При этом команды должны быть выражены в *машинном коде*, т.е. на *машинном языке* данного процессора. Каждая команда задает элементарное действие, такое как сложение, вычитание, умножение, сравнение и т.д., которое совершается процессором с очень большой скоростью – порядка миллиона в секунду, и более. Во время выполнения *программы* – списка команд, необходимого для работы компьютера, данные извлекаются (читаются) из оперативной памяти, а результаты заносятся (записываются) в оперативную память. В каждый отдельный момент времени в процессоре находятся только те данные, которые необходимы текущей выполняемой команде, т.е. очень небольшое количество операндов. Элементы внутренней памяти центрального процессора называются *регистрами*. Все данные, в которых нет непосредственной необходимости, находятся в оперативной памяти, играющей, таким образом, роль «камеры хранения».

С остальными устройствами компьютера, и в первую очередь с оперативной памятью, центральный процессор связан с помощью так называемых *шин* – системы параллельных проводников. К основным типам шин центрального процессора относятся:

1 **Адресная шина –** служит для указания процессором адреса ячейки оперативной памяти, с которой будет начинаться запись или считывание информации.

2 **Шина данных** – через которую осуществляется обмена информацией между оперативной памятью компьютера и регистрами центрального процессора.

3 **Шина команд** – используется для пересылки команд из оперативной памяти в специальный *регистр команд* центрального процессора. Самые простые команды укладываются в один байт, однако есть и такие, которые занимают два, три и более байта.

Шины центрального процессора имеют различную *разрядность* – т.е. различное количество параллельных проводников. Так, например, у наиболее распространенных процессоров семейства Intel Pentium адресная шина и шина команд – 32-разрядные, а данных – 64-разрядная. Большинство регистров общего назначения у процессоров семейства Intel Pentium также 32-разрядные. Емкость (разрядность) регистра общего назначения называется еще длиной *машинного слова*. Поэтому говорят, что процессоры семейства Intel Pentium 32-разрядные.

Длина машинного слова является следующей, после системы команд, по важности характеристикой центрального процессора, поскольку она определяет, сколько информации за одно обращение может быть переслано между оперативной памятью и процессором и, соответственно, обработано в нем с помощью одной машинной команды. При некоторых обстоятельствах эта характеристика может иметь решающее влияние на производительность всего компьютера и стать его «узким местом», так как в современных процессорах скорость обработки информация гораздо выше скорости ее обмена с оперативной памятью. На скорость же обработки информации внутри процессора влияет его *внутренняя тактовая частота*. В современных процессорах семейства Intel Pentium она достигает порядка нескольких ГГц. Для сравнения, процессор Intel 8086, на основе которого был построен первый компьютер IBM PC, работал на частоте всего лишь 4.77 МГц!

Оперативная память (или *оперативное запоминающее устройство* – **ОЗУ**, **RAM** – Random Access Memory – память произвольного доступа), как было сказано выше, предназначена для хранения программ и данных,

необходимых при их выполнении (предназначена для хранения выполняемой программы и принадлежащие ей данные). Организация оперативной памяти компьютера чем-то напоминает систему абонентских почтовых ящиков, с которыми мы имеем дело в повседневной жизни, в том смысле, что они содержат и хранят объекты. То есть, объекты содержатся во множестве отдельно адресуемых *ячеек памяти*, каждая из которых имеет свой персональный *адрес*. Каждое обращение к памяти должно сопровождаться указанием адреса соответствующей ячейки. Но на этом сходство и заканчивается.

Способность же оперативной памяти компьютера хранить данные основывается совсем не на том, что объекты физически присутствуют в ней, а на том, что состояние самой ячейки соответствует некоторому закодированному представлению объекта. Следовательно, ячейка должна принимать некоторое количество *дискретных состояний*. Трудно изготовить компоненты, способные принимать любое из множества отличных друг от друга состояний и оставаться в нем сколь угодно долго. Однако можно без особых затруднений строить запоминающие элементы, которые имеют только два различных состояния. Такие элементы называются *двоичными запоминающими элементами*. Если сгруппировать n двоичных запоминающих элементов, то эта группа может принимать 2^n различных состояний. Если эту группу рассматривать как некоторую неделимую компоненту, то она представляет собой ячейку памяти с 2^n возможными состояниями. Как было сказано выше, один двоичный разряд в такой группе, который может принимать одно из двух значений – 0 или 1, называется *бит*. Группа, состоящая из восьми (n = 8) двоичных разрядов (бит) называется *байт*. Таким образом, байт может принимать одно из $2^8 = 256$ различных значений – от 0 до 255.

Каждая ячейка оперативной памяти имеет свой адрес, который выражается целым положительным числом. Максимальный объем оперативной памяти компьютера в первую очередь зависит от разрядности адресной шины процессора. Так, в процессорах Intel Pentium с 32-разрядной адресной шиной

непосредственно можно обратиться к ячейке с максимальным адресом 2^{32} = 4 294 967 296, т.е. максимальный объем оперативной памяти компьютера построенного на этом процессоре не может превышать 4.3 Гбайт. Однако это отнюдь не означает, что именно столько памяти непременно должно быть в компьютере. Предельный размер оперативной памяти, которую можно установить в компьютере, иногда зависит от так называемого *чипсета* (*микропроцессорного комплекта*) – набора вспомогательных микросхем, управляющих обменом данных внутри компьютера – и не может превышать нескольких сот Мбайт. Помимо ограничений, накладываемых на размер оперативной памяти компьютера центральным процессором и другими вспомогательными микросхемами, он зависит также от характера задач, решаемых на компьютере, а также от ее удельной стоимости. Тогда, при выполнении всех этих ограничений, можно сформулировать следующее правило – *для компьютера общего назначения лишней памяти не бывает никогда*.

Объем информации, пересылаемый за одно обращение, между оперативной памятью и центральным процессором зависит от разрядности шины данных центрального процессора. Обычно это одно из значений следующего ряда: ***8***, ***16***, ***32***, ***64*** и т.д. байт. Из него видно, что *минимально адресуемой величиной оперативной памяти компьютера является байт*. При этом для процессоров семейства Intel Pentium за один раз пересылается 64 бита, или 8 байт, информации. Даже если необходимо переслать один байт информации, все равно пересылается 8. Такая процедура пересылки информации принята потому, что между центральным процессором и оперативной памятью находится буферная, так называемая, *кэш-память*, которая работает на частоте центрального процессора, и в подавляющем большинстве программ данные в оперативной памяти расположены подряд. При этом если данные уже находятся в кэш-памяти их пересылка в регистры центрального процессора (и, соответственно, из регистров в кэш-память) выполняется на несколько порядков быстрее, чем прямо из оперативной

памяти. За счет применения кэш-памяти и «широкого» считывания информации из оперативной памяти (и, соответственно, записи) значительно возрастает общее быстродействие компьютера.

При работе с оперативной памятью следует четко различать отличие между адресом ячейки (адресом байта) и содержанием ячейки (его значением). Адрес (номер) ячейки всегда один и тот же, а вот ее содержимое может меняться в диапазоне от 0 до 255. Это напоминает упоминавшуюся выше систему абонентских почтовых ящиков, адреса которых всегда остаются постоянными, а вот содержимое в каждый отдельный промежуток времени может быть совершенно различным.

Принципиальной особенностью оперативной памяти является ее способность хранить информацию только во время работы компьютера. В процессе выполнения программы содержимое ячеек памяти постоянно меняется – в одни из них информация заносится из процессора как результаты выполнения некоторых арифметических или логических операций, в другие информация пересылается из иных участков памяти, а в третьи – из внешней среды через *внешние устройства*. По выражению известного американского программиста Питера Нортона, оперативная память – это черновик, где временно записываются программы, данные, и результаты обработки – полуфабрикаты. Для пользователя же эти результаты обретают завершенный вид, если они оформлены и выведены в определенном виде на экран дисплея, на принтер, на звуковые колонки и т.д. После загрузки же в оперативную память новой программы, ее прежнее содержимое замещается новым, а после выключение компьютера – и вовсе пропадает.

Существует много разновидностей оперативной памяти, но с точки зрения физических принципов ее работы различают два основных вида:

1 DRAM – Dynamic RAM – динамическая память, и

2 SRAM – Static RAM – статическая память.

Ячейки динамической памяти (DRAM) представляют собой микроконденсаторы, которые на своих пластинах способны хранить

электрический заряд. Это наиболее распространенный, экономичный и доступный тип памяти. Однако этот тип памяти обладает и определенными недостатками, которые заключаются в следующем:

- Как при заряде, так и при разряде конденсаторов неизбежны переходные процессы, в связи с чем запись и считывание информации осуществляется в несколько раз медленнее, чем в памяти типа SRAM.
- С течением времени заряды ячеек имеют свойство рассеиваться в пространстве, причем весьма быстро, что может привести к потере информации. Для борьбы с этим явлением приходится постоянно *регенерировать* (*освежать*, *подзаряжать*) ячейки оперативной памяти, что является дополнительным источником непроизводительных расходов ресурсов компьютера, и как следствие – снижение быстродействия.

Ячейки статической памяти (SRAM) представляют собой электронный *триггер*, имеющий два устойчивых состояния. Поскольку в триггере хранится не заряд, а состояние (включен/выключен), то этот тип памяти обеспечивает более высокое быстродействие, хотя за счет того, что каждая ячейка содержит несколько транзисторов и других электронных компонентов, он сложнее и, соответственно, дороже.

Микросхемы динамической памяти в компьютерах общего применения используются, как правило, в качестве основной оперативной памяти, микросхемы же статической памяти – в качестве вспомогательной (промежуточной, *кэш*-) памяти, предназначенной для увеличения быстродействия компьютера.

Элементы динамической памяти для компьютеров общего применения конструктивно выполняются в виде так называемых *модулей*, которые вставляются в соответствующие разъемы шин. При этом они имеют два основных исполнения:

- SIMM – Single In-line Memory Module – модуль памяти с однорядным расположением выводов, и

- DIMM – Dual In-line Memory Module – модуль памяти с двухрядным расположением выводов.

На компьютерах с процессором Intel Pentium SIMM-модули можно применять только парами, в то время как DIMM-модули можно применять и по одному. Некоторые компьютеры имеют разъемы и для SIMM- и для DIMM-модулей, хотя не все из них допускают их одновременное использование.

Основными характеристиками модулей оперативной памяти является их емкость и время доступа. SIMM-модули, как правило, изготовляются емкостью 4, 8, 16 и 32 Мбайт, а DIMM-модули – 16, 32, 64, 128, 256 Мбайт и более. Время доступа показывает, сколько времени необходимо для обращения к ячейке памяти – чем оно меньше, тем быстрее может выполняться обмен с памятью. Время доступа измеряется в *наносекундах – нс*. Его типичное значение для SIMM-модуля – 50 ÷ 70 нс, а для DIMM-модуля – 7 ÷ 10 нс.

Шины – это среда передачи информации между всеми устройствами компьютера. Физически шина представляет собой ряд проводников (линий), разъемов (*слотов*) и электронных схем (чипсет), а логически – это определенный набор (семейство) правил обмена информацией. Каждое такое семейство правил имеет свое, уникальное, имя – название шины. Синонимом термина «шина» является «магистраль». Выделенная из общей шины группа линий, выполняющих одинаковые функции, также называется шиной. Это такие шины, как:

1 **шина данных**, предназначенная для передачи данных,
2 **адресная шина**, которая используется для адресации устройств компьютера и
3 **шина команд** (**управления**), служащая для передачи команд (управляющих сигналов).

Неоднозначностей между понятием общей шины и et составляющими, как правило, не возникает, поскольку смысл всегда понятен из контекста. Основная характеристика шины – ее пропускная способность, которая самым

существенным образом влияет на производительность всего компьютера в целом.

В персональных компьютерах общего применения различают два типа шин:

- **локальная**, обеспечивающая обмен между центральными устройствами компьютера, такими как процессор и оперативная память, и
- **расширения**, предназначенная для подсоединения внешних устройств и являющаяся производной от локальной.

Одной из первых, широко используемой, локальной шиной является Multibus1, первоначально разработанная фирмой Intel для компьютеров на базе процессора Intel 8086. Ее адресные шины мультиплексируются с шинами данных. Или, проще говоря, одни о те же линии попеременно используются для передачи адресов и данных. Позднее эта шина была доработана и для использования в компьютерах на базе процессора Intel 80286. Отечественным аналогом шины Multibus1 является шина И-41. Для компьютеров с 32-разрядным процессором, начиная с Intel 80386, используется локальная шина Multibus2, также разработанная фирмой Intel. Начиная с процессоров Intel Pentium Pro, используется локальная шина FSB – Front Side Bus. Et частота является одним из основных потребительских параметров компьютера, и может быть одной из ряда: 66, 100, 133, 200, 266 МГц и более.

С самой шиной расширения пользователь компьютера непосредственно дела не имеет. Он «общается» с ней посредством гнезд (разъемов, слотов) расширения, вставляя в них различные платы расширения – графические, звуковые, сетевые и т.д. Хотя за время, истекшее с момента появления первого компьютера, было разработано довольно много типов шин расширения, и связанных с ними слотов, все они разрабатывались в рамках определенных стандартов. Именно благодаря этой стандартизации слоты расширения стали открытой системой, допускающей быструю модернизацию и расширение функций компьютера. Уже при выпуске первых компьютеров фирма IBM сделала доступными для всех технические справочники с описанием

назначения контактов и всех необходимых технических деталей слотов расширения. Для фирм-производителей это явилось стимулом к разработке новых и более совершенных плат расширения. Этот процесс усовершенствования не прекращается и в настоящее время – пока не видно ни его конца, ни насыщения рынка.

Разнообразные современные шины расширения, разработанные с использованием последних достижений науки и техники, далеко ушли от своих предшественников. Но гнезда расширения, обеспечивающие работу этих шин, по своему функциональному назначению и внешнему исполнению, мало чем отличаются от стандартных слотов первых компьютеров фирмы IBM. При этом обычно не возникает никаких проблем, связанных с выбором требуемой платы расширения любого функционального назначения – нужно лишь выяснить, какая шина, или шины, используются в данном компьютере. В компьютерах семейства IBM PC наиболее широкое распространение получили следующие типы шин расширения:

PC-BUS – Personal Computer Bus – шина персонального компьютера (8-разрядная ISA, мини-ISA). Исторически это самая первая шина расширения семейства компьютеров IBM PC. Именно благодаря ей, не в последнюю очередь, эти компьютеры получили столь широкое распространение. Она предназначалась для одновременной передачи 8 бит информации и, кроме того, включала 20-разрядную адресную шину, что ограничивало адресное пространство компьютера пределом в 1 Мбайт. Для подключения плат расширения использовались специальные 62-контактные слоты. Заметим, что процессор и шина синхронизировались от одного тактового генератора с частотой 4.77 МГц. При этом максимальная скорость передачи данных не превышала 2 Мбайт/с.

ISA – Industry Standard Architecture – архитектура промышленного стандарта (AT-BUS, AT-шина). Следующее поколение персональных компьютеров было разработано на базе процессора Intel 80286, для использования всех возможностей которого необходима была новая шина с

большей разрядностью. В результате развития шины PC-BUS была создана новая шина расширения ISA, которая в дальнейшем оказала самое существенное влияние на дальнейшее развитее архитектуры персональных компьютеров семейства IBM PC. Прежнее 62-контактное гнездо 8-разрядной шины PC-BUS было расширено путем добавления дополнительного 36-контактного гнезда. На это дополнительное 36-контактное гнездо подавались все дополнительные сигналы, необходимые для функционирования новой шины.

Введение небольшого дополнительного гнезда и сохранение прежнего гнезда расширения обеспечило совместимость снизу вверх, т.е. возможность использования всех старых плат расширения. В результате увеличения гнезда расширения теперь стало возможным передавать параллельно 16 разрядов данных, а 24-разрядная адресная шина позволяла напрямую обращаться к 16 Мбайт оперативной памяти компьютера. Поскольку при использовании шины ISA обеспечивается совместимость снизу вверх, то компьютеры, в которых она используется, могут вообще не иметь ни одного 8-разрядного гнезда PC-BUS.

Компьютеры с шиной ISA уже допускали возможность синхронизации работы самой шины и процессора разными тактовыми частотами, что позволяло устройствам, выполненным на платах расширения, работать медленнее, чем базовый процессор. Это стало особенно актуальным, когда тактовая частота процессора превысила 10-12 МГц. Теперь шина ISA стала работать асинхронно с процессором на частоте 8 МГц. Максимальная пропускная способность ее при этом составила 5 Мбайт/с.

Казалось, что по мере все более широкого использования 32-разрядных процессоров (Intel 80386, Intel 80486, Intel Pentium) процесс модернизации шины ISA пойдет дальне. Однако на деле все оказалось совсем не так.

До сих пор все еще некоторые современные компьютеры оснащаются, как правило, тремя слотами расширения шины ISA. Естественно, что при использовании установленных в эти гнезда плат расширения в системе появляются «узкие места» – передача информации к плате расширения в 32-

разрядной системе вынуждена происходить через «узкое» 16-разрядное гнездо шины ISA. При этом потеря производительности компьютера очень сильно сказывается на работе тех плат расширения, для которых действительно важно очень высокое быстродействие. В первую очередь сюда относятся графические платы, контролеры жестких дисков и сетевые платы. Но, не смотря на столь низкую, по нынешним меркам пропускную способность, шина ISA продолжает использоваться и посей день в некоторых современных компьютерах для подключения сравнительно «медленных» устройств таких, например, как звуковые карты, модемы, последовательные порты и т.д.

MCA – Microchannel Architecture – архитектура «Микроканал». По вышеназванным причинам совершенно ясно, что при использовании 32-разрядных процессоров необходим новый тип шины. При этом, начиная с компьютеров Intel 80386, стандарты шин расширения отличаются большим разнообразием.

В 1987 году фирма IBM представила совершенно новую шину MCA. Первоначально разработанная для процессора Intel 80286 с 16-разрядной шиной данных, она не обеспечила сколь нибуть существенного улучшения шины ISA. Шина MCA стала действительно интересной в расширенном варианте для 32-разрядных процессоров. Появилось гнездо расширения с полной шириной в 32 разряда для шин данных и адреса. Для подключения плат расширения в этом гнезде используются 188 контактов, расположенных на расстоянии 1.27 мм друг от друга.

Отличительные особенности этой шины – существенное увеличение объема памяти и производительности компьютера, а также повышение степени оптимизации электронных компонентов схемы. Архитектура MCA позволяет устанавливать отдельный процессор на каждой плате расширения, который сам может управлять работой шины. Благодаря такой возможности эти процессоры могут работать независимо и параллельно от центрального процессора компьютера, что значительно повышает производительность всей системы.

Все эти усовершенствования привели к тому, что при использовании шины MCA скорость обмена данными между отдельными модулями системы может достигать 10 Мбайт/с. При обращении к оперативной памяти скорость передачи данных может достигать 32 Мбайт/с.

Еще одно преимущество этой шины – возможность автоматического конфигурирования плат расширения. Каждая плата расширения для шины MCA имеет собственный идентификационный номер, который обеспечивает ее распознавание системой и автоматическую настройку. Платы расширения для шины ISA обычно оснащаются так называемыми DIP-переключателями или джамперами (маленькими перемычками), с помощью которых они «подгоняются» к имеющейся системе в соответствии с прилагаемым руководством. Исключением являются платы, которые конфигурируются с помощью поставляемого вместе с ними программного обеспечения. Для исключения конфликтов между платами расширения при таком конфигурировании необходимо, чтобы обеспечивалась совместимость с уже имеющимися платами расширения.

Но не смотря на многочисленные аргументы, говорящие в ее пользу, шина MCA не получила широкого распространения из-за имеющихся у нее существенных недостатков.

- Во-первых, она полностью не совместима с шиной ISA, что не позволяет использовать уже имеющиеся ISA-платы расширения в компьютерах с шиной MCA.
- Во-вторых, MCA-платы расширения гораздо дороже аналогичных ISA-плат, что в первую очередь связано с высокой лицензионной платой, введенной фирмой IBM для сторонних производителей.

Все это не позволило ей занять соответствующую нишу на мировом компьютерном рынке – в основном она используется в составе семейства компьютеров PS/2, производимых фирмой IBM.

EISA – Extended ISA – расширенная ISA. В качестве альтернативы шине MCA восемь известных фирм-производителей компьютерного оборудования

(AST, Compaq, Epson, Hewlett-Packard, NEC, Tandy, Wyse и Zenith) в 1989 г. разработали менее эксклюзивную, но сравнимую по производительности, шину расширения и назвали ее EISA, которая явилась дальнейшим расширением шины ISA.

Основной замысел при ее разработке заключался в том, чтобы создать шину, которая не вносит никаких ограничений в работу 32-разрядной системы, но в то же время допускает установку стандартных ISA-плат. В результате появился «двухэтажный» слот, по своей форме и длине соответствующий ISA-гнездам.

В EISA-разъем компьютера, помимо, разумеется, специальных EISA-плат, можно вставлять либо 8-, либо 16-разрядные платы расширения, изначально разработанные для шин PC-BUS и ISA, соответственно. Это обеспечивается простым, но поистине гениальным решением – в каждом EISA-гнезде расширения ниже стандартных ISA-контактов расположены контакты для EISA-плат расширения с дополнительными функциями. Механическая конструкция гнезда не допускает ошибочного проскальзывания контактов ISA-плат к EISA-контактам. ISA-платы расширения устанавливаются только в верхней части колодки, а EISA-платы устроены так, что они могут достичь нижних контактов.

Шина EISA позволяет использовать 32-разрядные шины данных и адреса и, кроме того, предоставляет возможность управления со стороны плат расширения. При этом EISA-плата расширения может установить полный контроль над шиной, что обеспечивает оптимальные условия для ее работы. Возможна также автоматическая конфигурация EISA-плат расширения, а также автоматическое опознавание типа платы расширения, установленной в слот, – ISA или EISA.

Естественно, что полную производительность шина EISA реализует лишь при установке только 32-разрядных плат расширения EISA, а не «подтормаживающих» работу 16-разрядных ISA-плат или 8-разрядных PC-

BUS-плат. При одновременном использовании EISA-, ISA- и PC-BUS-плат расширения шина будет работать в наиболее медленном режиме.

По максимальной скорости передачи данных 33 Мбайт/с в так называемом *пакетном режиме* (*burst mode*) шина EISA даже превосходит шину MCA. При этом такая высокая скорость передачи достигается лишь на частоте шины 8.33 МГц.

VLB – VESA Local Bus – локальная шина VESA. Наиболее «узким» местом компьютеров, построенных на базе 32-разрядных процессоров Intel 80486 и шины EISA, стал обмен данными с графическими устройствами и жесткими дисками. Эту проблему в свое время решила шина VLB, предложенная ассоциацией VESA (Video Electronics Standards Association – ассоциация по стандартам в области видеоэлектроники). Поэтому неудивительно, что наиболее широко используемыми платами расширения для шины VLB являются именно графические платы. Но в качестве устройств расширения, подключаемых к этой шине, выступают также платы котроллеров жестких дисков и сетевые платы.

Конструктивно под плату расширения используется стандартный ISA- или EISA-разъем, работающий на тактовой частоте 8 МГц, в сочетании с дополнительным разъемом, в котором учтены все особенности стандарта VLB. Этот дополнительный разъем «врезан» в локальную шину компьютера, т.е. имеет непосредственный доступ к шинам центрального процессора и, соответственно, работал на его внешней тактовой частоте. В зависимости от используемого типа процессора тактовая частота шины VLB может изменяться от 20 до 66 МГц. При тактовой частоте 40 МГц шина VLB обеспечивает пиковую пропускную способность до 64 Мбайт/с, а при 50 МГц – 130 Мбайт/с.

Однако при всем этом VLB не является ни новым стандартом, ни новой ступенью в разработке шин персонального компьютера. При этом основным недостатком интерфейса VLB является то, что предельная частота шины и, соответственно, ее пропускная способность сильно зависит от числа подключенных устройств. Так, например, при частоте 50 МГц к шине можно

подключить лишь одно устройство, при частоте 40 МГц – два, а 33 МГц – три устройства.

PCI – Peripheral Component Interconnect – межсоединение периферийных компонентов. Новая скоростная шина расширения в основном была разработана фирмой Intel первоначально для компьютеров на базе процессоров Intel Pentium, хотя используется и в компьютерах на базе процессора Intel 80486.

По своей сути это тоже интерфейс локальной шины, связывающей процессор с оперативной памятью, в который «врезаны» разъемы для подключения внешних устройств. При этом обозначение Local Bus для этой шины, начиная с PCI спецификации 2.0, уже вообще не применимо. В данном случае локальная шина и шины расширения четко отделены друг от друга. Связь между ними осуществляется через так называемые мосты (bridges), функции которых выполняют вспомогательные микросхемы (чипсет). При этом за пакетную передачу данных отвечает не процессор, а мосты. Процессор может продолжать свою работу, когда шина PCI записывает данные в память или считывает их из нее. То же самое происходит и при обмене данными между двумя PCI-платами расширения. Благодаря этой возможности обеспечивается заметное повышение производительности в многозадачных режимах с помощью специально разработанных для этих целей программ-драйверов.

Эта шина поддерживает работу с 32-разрядными и 64-разрядными данными. Во втором случае шин адреса и данных мультиплексируются, т.е. одни и те же линии попеременно используются для передачи и адресов и данных. Переход на 64-разрядную шину приводит к гигантскому увеличению адресуемого пространства памяти – 2^{64} = 16.7 миллионов терабайт.

Развязка с помощью мостов локальной шины и шин расширения, а также «ширина» в 64 разряда обеспечивают гарантию того, что данная шина будет работать и с процессорами следующих поколений, и можно быть уверенным в том, что в ближайшем будущем не возникнет ни каких проблем с объемом адресуемой памяти. Фактически шина PCI заменила собой другие 32-разрядные

шины, такие как EISA и VLB. В тоже время она может одновременно применяться с 16-разрядной шиной ISA под широко распространенные 16- и 8-разрядные платы расширения.

Шины PCI спецификации 2.0 работает с тактовой частотой 33 МГц и обеспечивает пропускную способность до 234 Мбайт/с. Спецификации 2.1 поддерживает частоту 66 МГц и пропускную способность, соответственно, до 528 Мбайт/с. Разумеется, что PCI-плата расширения, рассчитанная на частоту 66 МГц, при ее установке в разъем с частотой 33 МГц будет работать на частоте шины – 33 МГц. Кроме того, гарантируется, что плата расширения с частотой 33 МГц, при ее установке в разъем с частотой 66 МГц, также будет работать на частоте 33 МГц. При этом следует учитывать, что единственная такая плата снизив рабочую частоту шины до 33 МГц, уменьшит, соответственно, общую производительность системы.

К стандартной PCI-шина могут подключаться до 10 слотов расширения, хотя в большинстве компьютеров, как правило, устанавливается не более трех таких разъемов. Это объясняется тем, что каждая PCI-плата расширения может разделяться между двумя периферийными устройствами, уменьшая, таким образом, общее число устанавливаемых разъемов.

Важным нововведением, реализованным в этом стандарте, стала поддержка так называемой технологии Plug-and-Play (Включай и Работай), впоследствии оформившейся в промышленный стандарт. Ее суть состоит в том, что после физического подключения внешнего устройства к разъему шины PCI происходит его автоматическая настройка на совместную работу с другими устройствами в составе компьютера. При этом, естественно, отпадает масса проблем, связанных с конфликтами между устройствами, имевшими место при их подключении к шине ISA. Но для того чтобы воспользоваться всеми преимуществами этой технологии необходимо, чтобы ее поддерживали не только шина, но также и плата расширения, BIOS (Basic Input/Output System – базовая система ввода-вывода) и операционная система.

Таким образом, PCI-шина представляет собой наиболее развитый стандарт. Наряду с 16-разрядными ISA-разъемами это один из наиболее стандартизованных компонентов современного компьютера.

AGP – Accelerated Graphics Port – ускоренный графический порт. Как при внедрении шины VLB, так и при внедрении шины PCI видеоадаптер всегда был первым устройством, «врезаемым» в новую шину. Однако сегодня параметры шины PCI уже не соответствуют требованиям видеоадаптеров, и поэтому для них была разработана отдельная шина, получившая название AGP. Ее тактовая частота соответствует тактовой частоте шины PCI, но она имеет более высокую пропускную способность – в режиме 4-х кратного умножения она достигает 1066 Мбайт/с.

PCMCIA – Personal Computer Memory Card International Association – международная ассоциация производителей плат памяти для персональных компьютеров IBM PC. Первые устройства, соответствующие этому стандарту задумывались как альтернатива относительно тяжелым и энергоемким приводам флоппи-дисков в портативных компьютерах. PCMCIA-устройства размером с обычную кредитную карточку являются альтернативой обычным платам расширения, подключаемым к шинам расширения. Сегодня в этом стандарте выпускаются модули памяти, модемы, сетевые карты, винчестеры и т.д. Особой популярностью пользуются PCMCIA-карты флэш-памяти, которые не теряют информацию при выключении питания, обладают высоким быстродействием и могут быть использованы в качестве винчестера без движущихся частей.

Для настольных компьютеров также разработаны адаптеры PCMCIA-устройств. Они представляют собой плату расширения, которая вставляется в обычный слот расширения и соединяется с разъемом PCMCIA ленточным кабелем. Сам PCMCIA-разъем размещается в стандартном отсеке с формат-фактором 3.5” или 5.25”.

USB – Universal Serial Bus – универсальная последовательная шина. Она определяет способ взаимодействия компьютера с периферийным

оборудованием и позволяет подключать до 256 различных устройств, имеющих последовательный интерфейс. При этом устройства могут включаться цепочками, т.е. каждое следующее устройство подключается к предыдущему. Производительность шины USB относительно не велика – не более 1.5 Мбайт/с, но для таких устройств, как клавиатура, мышь, модем, джойстик и т.п., этого вполне достаточно. Удобство шины состоит в том, что она практически исключает конфликты между различным оборудованием, позволяет подключать и отключать устройства в «горячем режиме», т.е. без выключения компьютера, а также объединять несколько компьютеров в простейшую локальную сеть без применения специального оборудования и программных средств.

Блокам УУ $ВУ_1$ – УУ $ВУ_n$ на рис. 1.4 соответствуют «устройству управления внешним $устройством_1$» ÷ «устройству управления внешним $устройством_n$». Устройства управления внешними устройствами (контролеры, карты, адаптеры, платы расширения) предназначены для управления работой внешних устройств. Как правило, каждому внешнему устройству соответствует свое устройство управления; хотя существуют и многоканальные контролеры, каждый из которых управляет работой несколько внешних устройств. Конструктивно устройство управления может быть выполнено как в виде небольшой интегральной микросхемы, так и в виде довольно сложной платы расширения. Так работой клавиатуры и мыши в персональном компьютере управляют простые контролеры, а монитором – достаточно сложное устройство – видеокарта (видеоадаптер).

Видеокарта совместно с монитором образует так называемую *видеосистему* персонального компьютера. Ввиду того, что эта система самым существенным образом влияет на потребительские свойства всего компьютера, параметры видеокарты рассмотрим более подробно.

Современный видеоадаптер включает в себя видеопамять, в которой хранится изображение, отображаемое на экране монитора, постоянное запоминающее устройство (ПЗУ), в котором записаны наборы шрифтов, а

также функции BIOS для управления им. Кроме того, он содержит и другие микросхемы, которые обеспечивают обмен данными с центральной частью компьютера, формирование изображения на экране монитора и т.д.

Видеоадаптер может работать в различных текстовых и графических режимах. В текстовых режимах экран монитора разделяется на отдельные текстовые позиции (*знакоместа*), в каждой из которых может отображаться один символ. Каждое такое знакоместо однозначно определяется двумя координатами: X – номер столбца и Y– номер строки. Начало координат (0, 0), как в текстовых, так и в графических режимах, находится в верхнем левом углу экрана. В стандартном текстовом режиме экран монитора разбивается на 25 строк по 80 позиций в каждой. Но существуют и другие текстовые режимы, например, с числом строк 30, 43, 50, 60 и числом позиций в них 40, 132 и т.д. Изображение отображаемого символа формируется на прямоугольной точечной матрице. Точки, образующие изображение символа, называются передним планом (foreground), а остальные – фоном (background). Для основных текстовых режимов используются матрицы 8×8, 8×14, 9×14, 9×16 точек и т.д. Разумеется, что чем больше размер матрицы, тем выше качество изображений символов. Формирование же точечной матрицы выполняется знакогенератором на основе кодов символов, каждому из которых однозначно соответствует «разложенное» по строкам его изображение. Эти, «разложенные» по строкам, изображения символов хранятся либо в ПЗУ видеоадаптера, либо поставляется соответствующей программой-драйвером.

Графические режимы иногда называют режимами с адресацией всех точек (All Points Addressable), поскольку имеется прямой доступ к каждой отдельной точке выводимого изображения. При этом изображение формируется из отдельных точек, каждая из которых может быть окрашена в тот или иной цвет. Каждая такая точка называется *пикселем* (pixel – picture element – (минимальный) элемент изображения). Основной характеристикой работы видеоадаптера в графическом режиме является *разрешающая способность*, т.е. количество минимальных элементов информации (пикселей) отображаемых на

экране монитора по горизонтали и по вертикали, например, 320×200, 640×480, 800×600, 1024×768, 1152×864, 1280×1024 точек, и т.д. *Количество воспроизводимых цветов* (*глубина цвета*, *цветовое разрешение*) является следующей важной характеристикой видеоадаптера, и показывает количество цветов, с помощью которых может отображаться каждый отдельный пиксель. Глубина цвета зависит от количества бит, отводимых для ее кодирования следующим образом:

Бит	Цветов	Режим
1	$2^1 = 2$	Черно-белый
2	$2^2 = 4$	
8	$2^8 = 256$	
16	$2^{16} = 65\,536$	High Color
24	$2^{24} = 16\,777\,216$	True Color
32	$2^{32} = 4\,294\,967\,296$	

Вместе с повышением требований к качеству отображаемого на экране монитора изображения развивались и видеоадаптеры. Так, для семейства персональных компьютеров IBM PC были разработаны следующие основные типы видеоадаптеров:

MDA (Monochrome Display Adapter) – первый графический адаптер семейства компьютеров IBM PC. Он мог отображать только алфавитно-цифровую информацию в 25 строках по 80 символов в каждой на монохромном дисплее. Каждая символьная позиция состояла из матрицы 9×14 точек. Ни возможности вывода графики, ни тем более изменения цветов предусмотрено не было.

HGC (Hercules Graphics Card) – полностью эмулировал адаптер MDA в текстовом режиме, а также обеспечивал разрешение 720×250 точек в графическом режиме на монохромном дисплее, который применялся с адаптером MDA.

CGA (Color Graphics Adapter) – обеспечивал отображение матрицы 8×8 точек при 16 цветах в текстовом режиме, и 4-х цветов при разрешении 320×200 или 640×200 пикселей при 2-х цветах в графическом режиме.

EGA (Enhanced Graphics Adapter) – полностью эмулирует работу MDA и CGA адаптеров, и обладает многими дополнительными возможностями. В частности, разрешающая способность в графическом режиме доведена до 640×350 пикселей, а число одновременно наблюдаемых на экране цветов составляет 16 из палитры в 64 цвета. Он также обеспечивал формирование более качественного текста – символьная позиция состоит из матрицы 8×14 точек.

VGA (Video Graphics Adapter) – включает все режимы предыдущих адаптеров и расширяет их на большее число цветов и более высокую разрешающую способность, что обеспечивает полную программную совместимость по принципу «снизу вверх». Максимальная разрешающая способность этого адаптера составляет 640×480 пикселей в графическом режиме, и 80×30 знакомест с матрицей 9×16 точек в текстовом режиме. Он может одновременно отображать 256 цветов из общей палитры в 256 К цветов.

SVGA (Super VGA) – является дальнейшим развитием стандарта VGA в направлении увеличения разрешающей способности и количества воспроизводимых цветов. Поскольку адаптеры SVGA никак не стандартизованы, то разные производители выпускают эти адаптеры с разным разрешением и разным количеством цветов. Наиболее типичными являются разрешения 800×600 и 1024×768 пикселей при 65 536 (High Color) или 16.7 млн. (True Color) воспроизводимых цветах. Вместе с этими адаптерами обязательно поставляются также программы-драйверы, которые обеспечивают их совместную работу с различным программным обеспечением.

Способность видеоадаптера отображать большое количество цветов с высоким разрешением тесно связана с объемом видеопамяти. Чем больше объем видеопамяти адаптера, тем больше цветов он сможет отображать и тем выше будет его разрешающая способность. Количество же памяти, необходимое для хранения изображения, в зависимости от разрешающей способности и глубины цвета вычисляется по следующей формуле:

$$V = \frac{X \times Y \times Z}{8 \times 1024 \times 1024},$$

где V – объем видеопамяти (Мбайт);

X – разрешение по горизонтали (пиксель);

Y – разрешение по вертикали (пиксель);

Z – разрядность кодирования цвета (бит).

Так, например, при разрешении 1024×768 пикселей и глубине цвета 65 536 (16 бит – High Color) цветов необходимо 1.5 Мбайт видеопамяти. А для того, чтобы отображать эти 1.5 Мбайт на экране монитора, например, со скоростью 60 раз в секунду необходимо, чтобы видеоадаптер обрабатывал информацию со скоростью 90 Мбайт/с. И это только для статического (неподвижного) изображения. Любые же другие операции потребуют дополнительного увеличения скорости.

Для увеличения производительности видеокарт используются так называемые видеоускорители (видеоакселераторы). Они берут на себя часть функций по построению изображения, разгружая при этом центральный процессор и, соответственно, уменьшая количество пересылаемой информации. Такими операциями являются: рисование линий и многоугольников, закрашивание определенным цветом указанных многоугольников, перемещение фрагментов изображения и т.д.

Различают два типа акселераторов: 2D (2-dimension – 2-х мерные) и 3D (2-dimension – 2-х мерные). Первые из них оптимизированы работе с «плоскими» изображениями, например, для офисных приложений. Вторые же ориентированы на работу с компьютерными играми и профессиональными программами обработки трехмерной графики. Обычно в этих случаях используют разные математические принципы ускорения графических операций, но существуют акселераторы, обладающие функциями и двухмерного и трехмерного ускорения.

Клавиатура является основным устройством ввода информации персонального компьютера. В зависимости от режима работы программы, которая воспринимает ввод с клавиатуры, вводимая информация может интерпретироваться либо как команды, которые необходимо выполнить, либо как данные, подлежащие обработке. Основными характеристиками клавиатуры являются:

1. количество клавиш и
2. их взаимное расположение.

Стандартная клавиатура персонального компьютера представляет собой «доску», на которой в несколько рядов расположено более 100 клавиш двух цветов (рис. 1.6):

- «белые» и
- «серые».

«Белые» клавиши предназначены для ввода алфавитно-цифровой информации, «серые» – выполняют служебные функции, такие как завершение или отмена ввода, смена регистра вводимых символов, смена языка ввода, перемещение позиции ввода (курсора) и т.д. В правом верхнем углу клавиатуры расположена панель индикации с тремя светодиодными индикаторами – **Num Lock**, **Caps Lock** и **Scroll Lock**, которые показывают ее текущее состояние (режим работы).

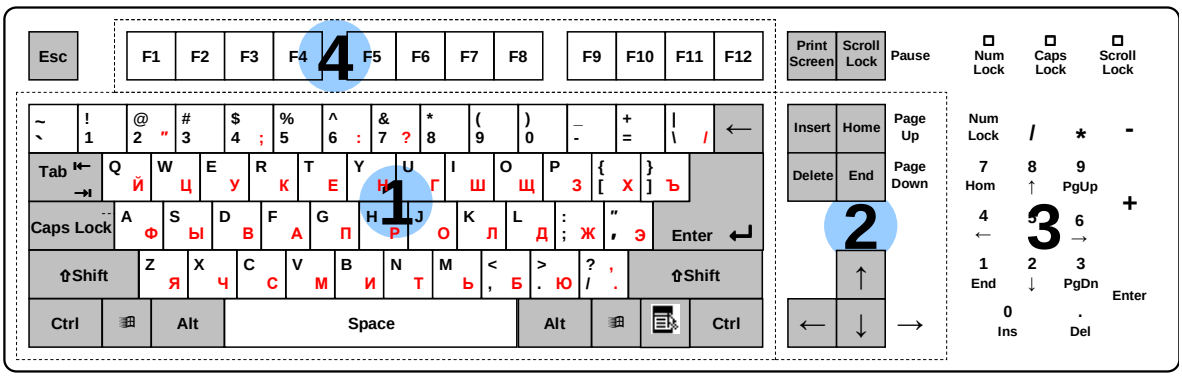


Рис. 1.6 – Стандартная клавиатура персонального компьютера

Визуально клавиши на клавиатуре распределены по следующим функциональным группам:

1. основная группа клавиш,

2. клавиши управления курсором,
3. дополнительные клавиши
4. функциональные клавиши и
5. специальные клавиши.

Однако такая классификация в действительности является довольно-таки условной, в том смысле, что в зависимости от выполняющейся в данный момент программы, любая клавиша может быть использована в совершенно произвольном качестве или, вообще, заблокирована. Поэтому в описании каждого программного продукта обязательно имеется раздел с изложением особенностей использования клавиатуры. Далее же будет дан обзор только наиболее универсальных приемов работы с клавиатурой, которые используются в большинстве программных средств.

Основную группу клавиш составляют алфавитно-цифровые и «серые» клавиши управления вводом. С помощью алфавитно-цифровых клавиш осуществляется ввод символьной информации: цифр, букв (русского и латинского алфавита, прописных и строчных), знаков препинания, символов арифметических операций и т.д. При этом каждая клавиша может использоваться для ввода нескольких символов. Для разных языков ввода существуют различные схемы закрепления символов национальных алфавитов за конкретными алфавитно-цифровыми клавишами. Каждая такая схема закрепления символов за конкретными клавишами называется *раскладкой клавиатуры*. Наиболее распространенные раскладки клавиатур для компьютеров уходят своими корнями в раскладки клавиатур для пишущих машин. Раскладки клавиатур принято именовать по символам, закрепленным за первыми слева клавишами верхней буквенной строки алфавитно-цифровой группы. Так, раскладка стандартной клавиатуры для персональных компьютеров типа IBM PC называется QWERTY. Наиболее часто используемым символом во всех раскладках клавиатур является пробел (Space), который вводится с помощью самой длинной клавиши, расположенной внизу алфавитно-цифровой группы. Очередной введенный символ эхом отображается

в соответствующем поле ввода на экране монитора, а сама позиция ввода обозначается *курсором* – вертикальной мигающей черточкой (▏). В исходном состоянии клавиатура, как правило, настроена на ввод прописных (нижнего регистра, малых) букв национального алфавита.

Самая большая «серая» клавиша **Enter** (**Return**, **Ввод**), расположенная справа от алфавитно-цифровой группы, служит для подтверждения момента завершения ввода. При этом если вводилась команда – она начнет немедленно выполняться, а если данные – будет закончен текущий абзац и начет следующий – курсор перейдет в начало следующей строки. Клавиша **Enter** используется также для подтверждения выбора пункта меню или выделенной кнопки графического интерфейса программы.

Клавиша **Esc**, которая принадлежит к специальной группе клавиш, и которая расположена в верхнем левом углу клавиатуры, используется для отмены вводимой информации. В большинстве современных операционных систем (и Windows в том числе) используется так называемый *буферизованный ввод*. Это значит, что вводимая с клавиатуры информация в начале «накапливается» в специально выделенном участке оперативной памяти компьютера – *буфере клавиатуры* – а лишь затем, по нажатию клавиши **Enter**, пересылается в программу, либо отбрасывается – по нажатию клавиши **Esc**. Клавиша **Esc** в этом смысле выступает в качестве некоторого антипода клавиши **Enter**. Поскольку буфер клавиатуры – это участок оперативной памяти вполне определенного размера, то когда-нибудь он может все-таки переполниться. Признаком того, что буфер клавиатуры переполнился, является характерный «писк» компьютера при вводе, а сами символы перестают эхом отображаться на экране монитора. Клавиша **Esc** используется также для возврата назад в цепочке выполненных ранее операций или выбора кнопки **Отмена** в графическом интерфейсе программы.

Клавиша **Caps Lock**, которая расположена слева от алфавитно-цифровой группы, служит для перевода клавиатуры в режим ввода прописных (верхнего регистра, больших) букв. Такое переключение называется фиксированным, или

жестким. О том, что клавиатура находится в режиме ввода прописных букв можно судить по свечению светодиодного индикатора **Caps Lock** в правом верхнем углу клавиатуры.

Две клавиши **Shift**, расположенные слева и справа алфавитно-цифровой группы, служат для временной смены регистра ввода букв (строчных на прописные, или прописных на строчные). Ввод букв при этом выполняется при удержании в нажатом состоянии одной из клавиш **Shift**. *Цифры всегда вводятся в нижнем регистре (как строчные), независимо от языка и положения переключателя* **Caps Lock**.

Клавиши **Alt** и **Ctrl**, расположенные слева и справа от клавиши пробела, самостоятельного значения не имеют. Они появились на определенном этапе развития компьютерной техники как одно из вынужденных и, надо признать, довольно-таки удачных технических решений. Дело в том, что в это время, для того чтобы управлять компьютером имеющихся клавиш стало не хватать, и необходимо было либо увеличивать их число и, соответственно, увеличивать саму клавиатуру (до размеров рояля!), либо искать какие-то другие пути решения этой проблемы. И поэтому было предложено вместо существенного увеличения количества клавиш добавить только две дополнительные клавиши, и использовать их для обеспечения необходимой функциональности клавиатуры в сочетании с другими клавишами.

Запись типа <клавиша$_1$>+<клавиша$_2$> означает, что вначале необходимо нажать <клавищу$_1$>, а затем, не отпуская ее, нажать <клавишу$_2$>.

Наиболее часто такое сочетание клавиш используется для смены языка ввода – русского на английский, английского на русский, русского на украинский и т.д. Поскольку такое переключение выполняется программным способом и, соответственно, является одной из функций операционной системы, то оно зависит от того, под управлением какой операционной системы работает компьютер. Так, например, в операционной системе Windows наиболее часто

переключение языка ввода выполняется с помощью сочетания клавиш **Ctrl+Shift**.

В большинстве программ не имеет никакого значения, какая из двух парных клавиш – **Alt**, **Ctrl** или **Shift** – была задействована. Однако некоторые программы делают такое различие между «левыми» и «правыми» (относительно пробела) клавишами. В таких случаях в справочной системе соответствующей программы (или документации к ней) даются явные указания относительно их использования.

Клавиша **Tab** используется для ввода неотображаемого символа табуляции и перемещения курсора на несколько позиций вправо. За каждым таким символом табуляции «закреплено» некоторое «расстояние» от левого края поля ввода. То есть, с помощью этой клавиши можно подготовить текст, выровненный, как в таблице, по колонкам. Она используется также для указания следующего по порядку элемента графического интерфейса пользователя. Для «обхода» элементов управления в обратном порядке используется сочетание клавиш **Shift+Tab**.

Клавиши  **Windows**, которые расположены слева и справа от клавиши пробела, служат для открытия или сворачивания основного меню операционной системы Windows.

Клавиша  **Context**, расположенная справа от клавиши пробела, служит для открытия контекстного меню выделенного элемента графического интерфейса пользователя операционной системой Windows.

Клавиша **BackSpace**, которая находится справа в самом верхнем ряду алфавитно-цифровой группы, над клавишей **Enter**, и которая довольно часто маркируется стрелкой, направленной влево (←), служит для удаления символа слева от курсора. Положение курсора и, соответственно, позиция редактирования при этом смещается влево.

Клавиша **Delete**, расположенная в группе клавиш управления курсором, предназначена для удаления символа справа от курсора. После удаления символа положение позиции редактирования остается неизменным.

Клавиши управления курсором служат, в основном, для управления курсором и, соответственно, позицией ввода и редактирования информации. Четыре клавиши со стрелками (←, →, ↑, ↓) перемещают курсор в указанном направлении на один символ.

Клавиши **Page Up** и **Page Down** служат для перемещения курсора на одну «экранную страницу» вверх или вниз, соответственно. Понятие «экранная страница» относится обычно к содержимому текущего окна. При этом с помощью клавиш **Page Up** и **Page Down** осуществляется «прокрутка» содержимого такого окна. Действие этих клавиш в конкретной программе может быть изменено с помощью соответствующего сочетания их с такими клавишами как **Alt** или **Ctrl**. Конкретный результат такой модернизации зависит от используемой программы.

Клавиши **Home** и **End** перемешают курсор в начало или конец строки, соответственно. Они могут использоваться также для перемещения на начало или конец некоторого списка. Их действие в каждой конкретной программе также может быть изменено с помощью сочетания с клавишами **Alt** или **Ctrl**.

Клавиша **Insert** традиционно служит для изменения режима ввода – *вставки* на *замену*, или наоборот – замены на вставку. Если курсор находится внутри текстовой строки, то в режиме вставки происходит ввод новых символов без замены существующих – текст как бы раздвигается вправо. В режиме же замены новые символы замещают уже существующие – как бы записываются «поверх» имеющихся. В исходном состоянии клавиатуры, как правило, используется режим вставки.

В некоторых программах функции клавиши **Insert** могут быть изменены. Возможно, также, что действие этой клавиши является настраиваемым. Поэтому, более подробную информацию следует искать в справочной системе конкретной программы.

Дополнительные клавиши (**вспомогательные клавиши**, **«правая клавиатура»**, **«дополнительная цифровая клавиатура»** и т.д.) дублируют некоторые клавиши основной клавиатуры. Этим блоком клавиш наиболее часто

пользуются операторы информационных систем, которые набирают, в основном, числовую информацию. Цифровые клавиши здесь совмещены с клавишами управления курсором. Нажатие же, расположенной в этой же группе, клавиши **Num Lock** переводит «белые» клавиши «правой клавиатуры» либо в режим ввода цифр, либо, наоборот, в режим управления курсором. При этом если светится индикатор **Num Lock**, то клавиатура находится в режиме ввода цифр, а если нет – в режиме управления курсором. Под рукой имеются также и все остальные клавиши, необходимые для ввода чисел – «точка» (.), которая используется в информационных системах для разделения целой и дробной частей числа вместо запятой, а также знаки арифметических операций: «серый» плюс, «серый» минус, «серый» знак умножить, «серый» знак разделить. Кроме того, здесь же находится и «двойник» клавиши **Enter** – «серый» **Enter**, – нажатием которой завершается ввод числа. При выключенном переключателе **Num Lock** клавиши дополнительной группы могут использоваться также для управления некоторыми компьютерными играми и дублирования функций *манипулятора мышь*.

Клавиши дополнительной группы выполняют еще одну очень важную функцию – ввод произвольных символов кодовой таблицы, особенно тех, за которыми не закреплены клавиши клавиатуры. Так, например, известно, что код символа «авторское право» (©) имеет значение 0169, а код символа «охраняемый знак» (®) – 0174, но соответствующих им клавиш на клавиатуре нет. Тогда для ввода таких символов поступают следующим образом:

1. Включить переключатель **Num Lock**.
2. Нажать и удерживать в нажатом состоянии клавишу **Alt**.
3. Не отпуская клавиши **Alt**, набрать на правой клавиатуре код требуемого символа, например, 0169 для символа «авторское право».
4. Отпустить клавишу **Alt**.

Символ, соответствующий набранному коду, появится экране в позиции ввода.

Функциональные клавиши от **F1** до **F12** расположены в верхней части клавиатуры. Они используются для выполнения наиболее часто употребляемых

команд. Конкретные команды, закрепленные за этими клавишами, зависят от работающей в данный момент программы, а в некоторых случаях, и от операционной системы. Общепринятым для большинства программ является соглашение о том, что клавиша **F1** служит для вызова справочной системы по данной программе.

Специальные клавиши – это клавиши, не вошедшие ни в одну из приведенных ранее групп, и которые расположены слева и справа от функциональных клавиш. Они выполняют специфические функции, целиком и полностью зависящие от операционной системы. Так, общепринятыми для операционной системы Windows являются следующие действия:

Esc – отмена только что выполненных действий. Более подробно ее использование было рассмотрено выше.

Print Screen (**SysRq**) – сохранение содержимого экрана в *буфере обмена* – специальной области оперативной памяти компьютера. Сочетание клавиш **Alt** + **Print Screen** выполняет сохранение в буфере обмена не всего экрана, а лишь активного окна.

Scroll Lock – используется некоторыми (как правило устаревшими) программами для фиксации курсора на одном месте и «пролистывания» всего документа – так называемый режим «скроллинга». Если этот режим включен, то светится светодиодный индикатор **Scroll Lock** на панели индикации.

Pause (**Break**) – приостановка или прерывание текущей выполняющейся программы.

При длительном удержании в нажатом состоянии большинства клавиш через некоторый промежуток времени начинается автоматический повтор их ввода. Параметры этого процесса, т.е. скорость повторения и интервал времени между нажатием клавиши и началом повторения ввода, могут быть настроены под индивидуальные особенности конкретного пользователя. Но, поскольку работой клавиатуры целиком и полностью управляет операционной системы, то более подробно эти средства будут рассмотрены в соответствующем разделе, посвященном настройке операционной системы. При этом можно настраивать

не только скорость повтора и время его начала, но и добавлять и удалять используемые языки ввода, способы переключения между ними, скорость мерцания курсора и т.д.

Рассмотренная стандартная клавиатура персонального компьютера IBM PC не является оптимальной во всех отношениях и имеет свои, специфические, недостатки:

1. она занимает много места,
2. не всегда удобна, и
3. раскладка клавиш далека от оптимальной.

Первый недостаток существенен в портативных (notebook) компьютерах и там он успешно решен. Второй недостаток преодолевается с помощью так называемых *эргономических* клавиатур. У таких моделей клавиатура приняла «волнистую» форму, в которой алфавитно-цифровой блок разделена на две части, расположенные под углом ~120°. Соответственно перенесен также блок клавиш управления курсором. Такие клавиатуры наиболее целесообразно применять, если необходимо вводить большие объемы алфавитно-цифровой информации. Они не только повышают производительность ввода информации и уменьшают общее утомлении пользователя, но и снижают вероятность и степень развития ряда заболеваний, таких как туннельный синдром кистей рук и остеохондроз верхнего отдела позвоночника.

Раскладки клавиатур берут свое начало от ранних образцов механических пишущих машин. Не существует никаких технических проблем изготовления клавиатур с любой раскладкой клавиш. Однако практическое использование таких клавиатур затруднено в связи с тем, что работе с ними необходимо специально учиться. Подобные клавиатуры используются лишь в ограниченном числе специальных случаев применения компьютеров.

Манипулятор мышь в качестве стандартного устройства персонального компьютера IBM PC появился гораздо позже других внешних устройств. Однако выполнение основных операций по управлению графическим интерфейсом пользователя большинства современных операционных систем

рассчитано именно на работу с мышью, а не с клавиатурой. В связи с тем, что мышь не была предусмотрена в составе первой, базовой, конфигурации персонального компьютера, для ее нормального функционирования необходима специальная программа-драйвер. И поэтому она доступна для работы не сразу после включения компьютера, а лишь по завершению процесса загрузки операционной системы.

Мышь относится к устройствам управления манипуляторного типа. Внешне она представляет собой небольшой плоский корпус с закругленными краями и двумя-тремя кнопками сверху, который с помощью тонкого провода подключается к компьютеру. Именно за такое внешнее сходство со взрослой мышью (примерно такие же размеры и такой же длинных хвост) эти манипуляторы также стали называться «мышами». Перемещение мыши по плоской поверхности синхронизировано с перемещением по экрану монитора специального графического объекта – *указателя мыши* (*mouse pointer*).

Форма указателя мыши зависит от:

1. объекта, на который он указывает,
2. состояния этого объекта и
3. выполняемой операции.

При этом наиболее распространенными формами указателя мыши в операционной системе Windows являются:

 – стандартная форма указателя – служит для указания острием стрелки на объект графического интерфейса пользователя, или точку на экране монитора.

 – для индикации того, что приложение выполняется в фоновом режиме, и необходимо дождаться завершения текущей операции, чтобы оно снова начало реагировать на действия пользователя.

 – для указания того, что система недоступна – она занята выполнением некоторой другой работы.

 – для индикации того, что в текущем состоянии данная операция невозможна.

 – для указания объекта графического интерфейса пользователя, относительно которого необходимо получить справочную информацию. Активизируется, обычно, по нажатию кнопки  **Справка** в окне диалога.

 – для обозначения того, что в графическом режиме далее будет производиться выделение объектов.

 – для указания того, что далее можно изменять границы выделенного объекта.

 – для обозначения того, что далее можно перетаскивать выделенный объект.

 – для выбора ссылки.

 – для обозначения режима рукописного ввода текста.

 – для указания того, что указатель мыши находится в области текстовой информации.

 *При работе в операционной системе Windows, и ее приложениях, необходимо четко отличать* ***текстовый курсор*** *от рассмотренного выше* ***указателя мыши****. Визуально текстовый курсор – это вертикальная мигающая черточка (**), которая показывает, что в данной позиции можно редактировать информацию. Он появляется лишь после того, как требуемая позиция будет обозначена с помощью указателя (*  *), и произведен одинарный щелчок левой кнопкой мыши. Указатель же мыши присутствует на экране монитора всегда.*

Как устройство ввода мышь фиксирует следующую информацию:

1. вид происшедшего события – т.е. какая именно кнопка была нажата, или отпущена, и
2. прямоугольные координаты точки на экране монитора, связанные с этим событием.

Каких-либо иных функций мышь не выполняет.

В большинстве случаев мышь имеет две кнопки. На левую кнопку ложится указательный палец, на правую – средний. И поскольку большинство людей являются «правшами», и держат мышь правой рукой, то основной кнопкой является левая. Для «левшей» же необходимо просто «поменять местами» функции левой и правой кнопок, что выполняется совершенно легко – с помощью указания соответствующих параметров программы-драйвера.

Назначение кнопок мыши, в общем случае, зависят от выполняющейся в данный момент программы. Однако общепринятым является следующее:

левая – основная – соответствуют клавише **Enter** на клавиатуре,

правая– вспомогательная – соответствуют клавише **Context** на клавиатуре.

Кроме двух, непременных, кнопок мышь может содержать и третью – среднюю, дублирующую в основном функции клавиши **Esc** на клавиатуре.

Основными технологическими операциями по управлению объектами графического интерфейса пользователя, которые выполняются при помощи мыши, являются:

1. Перемещение острия стрелки указателя мыши на объект – указать (на) требуемый объект.
2. Одинарный щелчок – быстро нажать и отпустить левую кнопку мыши – используется для указания (выделения, пометки и т.д.) требуемого объекта, необходимого пункта меню и т.д.
3. Двойной щелчок – два раза подряд быстро нажать и отпустить левую кнопку мыши – «открыть» для просмотра («войти», «запустить на выполнение») указанный объект. Необходимо четко отличать двойной щелчок от двух подряд идущих щелчков. В первом случае интервал времени между ними должен быть гораздо меньше, чем во втором – иначе они будет восприняты как два отдельных щелчка.
4. Щелчок правой кнопкой мыши – вызывает активизацию контекстного меню, т.е. такого меню, состав которого зависит от объекта, на котором был произведен щелчок;

5. Удерживая в нажатом состоянии левую кнопку мыши, переместить ее:
   а) если указатель был на «пустом» месте – будет нарисована контурная рамка, с помощью которой можно выделить несколько подряд идущих объектов, или создан некоторый рисованный объект;
   б) если указатель показывал на один или несколько выделенных объектов – они будут перемещены (транспортированы, перетащены) в позицию отпускания кнопки мыши – метод Перетащить и Оставить (Drag-and-Drop). После завершения перетаскивания с помощью *правой* кнопки мыши активизируется контекстное меню, с помощью которого можно уточнить выполняемую операцию – например, копирование или перемещение объекта.

Далее, при изложении материала, если не указано иное, будут использоваться следующие, общепринятые в компьютерной литературе, значения терминов:

1. «щелчок» – одинарный щелчок *левой* кнопкой мыши;
2. «нажатие» – нажатие и отпускание указанной *клавиши* на клавиатуре;
3. «двойной щелчок» – двойной щелчок *левой* кнопкой мыши;
4. «клавиша» – находится на клавиатуре;
5. «кнопка» – объект мыши;

В других же случаях специально будут указаны дополнительные, уточняющие, параметры выполнения необходимых операций.

Для работы с последними версиями операционных систем Windows рекомендуется использовать мышь IntelliMouse™, которая отличается от других устройств этого типа наличием небольшого колеса, расположенного между правой и левой кнопками. Наиболее эффективно с ее помощью выполняются такие операции, как:

1 Прокрутка текста в окне. Для этого необходимо вращать колесо в нужном направлении. Для непрерывной прокрутки текста нужно, удерживая в нажатом положении колесо, перемещать указатель мыши в соответствующем направлении.

2 Изменение масштаба отображаемого документа – удерживая нажатой клавишу **Ctrl** на клавиатуре, вращать колесо вперед или назад.

3 Переход по ссылке. При выполнении этой операции необходимо указать на любую ссылку и, удерживая нажатой клавишу **Shift** на клавиатуре, и прокручивать колесо назад для перехода к предыдущей ссылке или вперед – для перехода «вперед» по ссылкам.

В настоящее время наиболее распространенными конструкциями мышей являются:

- оптико-механические и
- оптические.

При перемещении оптико-механической мыши по горизонтальной поверхности вращается находящийся в ее нижней части шарик. Вращение этого шарика передается двум, взаимно перпендикулярным осям, на концах которых находятся колесики с регулярно расположенными отверстиями. При помощи специальных электронных элементов – оптронов – с колесиков считывается относительное перемещение мыши. В связи с тем, что шарик мыши делается из металла и обтягивается резиной, он плохо двигается по полированной или, наоборот, загрязненной поверхности. Поэтому для устранения «проскальзывания» под оптико-механическую мышь необходимо подкладывать специальный коврик.

Оптическая мышь механических частей не содержат. При ее перемещении по горизонтальной поверхности происходит излучение и прием отраженного от этой поверхности луча света. По изменению отраженного луча света можно судить об относительном перемещении оптической мыши. Этот тип мыши меньше чувствителен к таким механическим свойствам поверхности как чистота и шероховатость, однако более чувствителен к ее оптическим свойствам. С оптической мышью, также как и с оптико-механической, необходимо использовать специальный коврик. Однако теперь он служат другой цели – обеспечить оптическую неоднородность поверхности. И поэтому

иногда простой поворот такого коврика на 90° может привести мышь в нерабочее состояние.

Рассмотренные выше двух- и трехкнопочные мыши, а также с колесом, оптико-механические и оптические, далеко не являются исчерпывающими представлением всех существующих модификаций этого устройства – по сравнению с клавиатурой они имеют гораздо больше различных конструкциями. Так для некоторых специальных применений, таких, например, как САПР (Системы Автоматизированного Проектирования, CAD – Computer Aided Design), могут использоваться мыши с несколькими десятками кнопок. Однако наибольшее распространение получили все-таки оптико-механические двухкнопочные мыши, а в последнее время – еще и с колесом.

Поскольку мышь, как и любое другое устройство компьютера, управляется при помощи операционной системы, то большинство ее параметров можно подстроить под нужды и предпочтения конкретного пользователя. К числу таких регулируемых параметров относятся:

- скорость выполнения двойного щелчка,
- функции, закрепленные за левой и правой кнопками,
- форма указателя, в зависимости от выполняемой операции,
- скорость перемещения указателя мыши при заданном ее горизонтальном перемещении,
- и другие.

В случае необходимости предусмотрена также возможность «передачи» функций мыши клавиатуре. Более же детально процедура настройки различных параметров мыши будет рассмотрена в соответствующем разделе, вместе с другими настройками операционной системы.

Кроме обычной мыши в состав компьютера могут входить и другие манипуляторы, например:

- Джойстик – рычажно-нажимное устройство, которое используется, в основном, для управления компьютерными играми, и которое подключается, как правило, к специальному разъему на звуковой карте.

- Трекбол – в отличие от мыши устанавливается стационарно, и его шарик приводится в движение ладонью руки. Его основное преимущество заключается в том, что он не нуждается в специальной рабочей поверхности, и поэтому применяется в основном в портативных компьютерах.
- И некоторые другие.

Монитор (*дисплей*) персонального компьютера предназначен для вывода текстовой и графической информации. От его параметров во многом зависит производительность и комфортность работы на компьютере. По способу получения изображения наиболее распространенными являются следующие две группы мониторов:

1. на основе электронно-лучевой трубки (ЭЛТ) и
2. LCD – Liquid-Crystal Display – жидкокристаллический (ЖК) дисплей.

Наиболее распространенными мониторами для настольных персональных компьютеров являются мониторы, построенные на основе *электронно-лучевой трубки*. Они очень похожи на обычные бытовые телевизоры. У них тот же принцип формирования изображения – направленный электронный пучок вызывает свечение точек на экране дисплея. Этот тип мониторов позволяет создание изображения с максимальной яркостью, контрастностью и насыщенностью цветов.

Электронно-лучевая трубка – это вакуумная колба, внутри которой расположены:

1 одна (для черно-белого монитора) или три (для цветного) электронные пушки,
2 отклоняющая система,
3 модулятор и
4 экран, изнутри покрытый люминофором.

Принцип работы электронно-лучевой трубки заключается в том, что испускаемый электронной пушкой пучок электронов («электронный луч») попадая на люминофор экрана, вызывает эго свечение. На пути электронного луча расположена отклоняющая система, которая изменяет его направление, и

модулятор, регулирующий эго интенсивность. Изображение формируется при этом за счет перемещении электронного луча слева направо по горизонтальным линиям, которые называются *строками развертки – прямой ход*. По мере движения электронного луча по строке развертки видеосигнал, подаваемый на модулятор, меняет его интенсивность, за счет чего меняется интенсивность свечения люминофора в соседних точках, т.е. строится изображение. Когда луч доходит до правой стороны экрана он гасится и перемещается на начало следующей строки развертки, находящейся под предыдущей – *горизонтальный обратный ход*. После того, как луч таким образом «пробежит» по всем строкам экрана, он гасится и перемещается в левый верхний угол экрана – *вертикальный обратный ход*. Затем весь процесс повторяется снова. Отклонение луча по горизонтали в течение прямого хода осуществляется сигналом *строчной* (*горизонтальной*) развертки, а по вертикали – *кадровой* (*вертикальной*) развертки, которые подаются на отклоняющую систему ЭТЛ. След от луча на экране образует так называемый *растр* (рис. 1.7), и поэтому такие дисплеи называют еще растровыми.

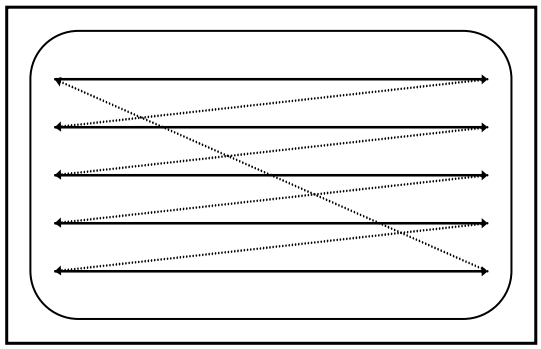


Рис. 1.7 – Растр электронно-лучевой трубки

В случае цветного монитора ЭТЛ имеет три отдельно управляемые электронные пушки, а поверхность экрана покрыта люминофором, состоящим из троек точек, соответствующих трем основным цветам:

R – Red – красный,

G – Green – зеленый и

B – Blue – синий.

Таким образом, каждая пушка должна «стрелять» только по точкам своего цвета.

Основными потребительскими параметрами монитора являются:

1. размер экрана по диагонали,
2. расстояние между точками люминофора,
3. частота кадровой (вертикальной) развертки,
4. частота строчной (горизонтальной) развертки,
5. ширина полосы частот видеосигнала и
6. соответствие стандарту безопасности.

Размер экрана монитора по диагонали принято измерять в дюймах. Для персональных компьютеров семейства IBM PC выпускаются мониторы следующего ряда стандартных значений: 8″, 14″, 15″, 17″, 19″, 21″ и 23″. Для наиболее популярных из них существуют свои оптимальные разрешения:

14″ – 640×480,

15″ – 800×600,

17″ – 1024×768,

19″ – 1280×1024.

Эти значения были выбраны исходя из того, что чем выше разрешение, тем больше информации можно отобразить на экране монитора, но тем меньше размер каждого отдельного элемента информации (пикселя) и, соответственно, меньше видимый размер отображаемых с помощью таких пикселей объектов. То есть, использование завышенного разрешения на мониторе малого размера приводит к тому, что объекты изображения становятся неразборчивыми, и работа с ними вызывает повышенное напряжение глаз, что приводит к их преждевременному утомлению. Использование же заниженного разрешения приводит к тому, что объекты изображения становятся крупными и, соответственно, их мало помещается на экране, что, в свою очередь, приводит к снижению производительности труда.

Расстояние между точками определяет удаление друг от друга соседних точек люминофора одинакового цвета. Чем меньше эта величина, тем четче

выглядит изображение. Расстояние между точками определяет верхний предел разрешения, которое может обеспечить данный монитор без потери качества. Так, например, для монитора с диагональю 17″ и соотношением сторон 3:4 длина большей стороны будет равна 326 мм. Тогда, для того чтобы он мог качественно воспроизводить изображение, например, с разрешением 1024 × 768 пикселей расстояние между соседними точками должно быть не более 326 : 1024 = 0.32 мм.

Частота кадровой (вертикальной) развертки определяет, как часто в течение секунды обновляется (регенерируется) изображение на экране монитора. Если, например, указывается частота 60 Гц, то это значит, что в течение одной секунды изображение «перерисовывается» 60 раз. Однако, 60 Гц – это довольно-таки посредственное значение для рассматриваемого параметра. При такой частоте регенерации изображение нельзя считать устойчивым (без заметного мерцания). Чем выше это значение, тем меньше мерцание экрана монитора и, следовательно, меньше нагрузка на зрение. Сегодня минимально допустимым значением частоты кадровой развертки считается значение 75 Гц, нормальным – 85 Гц, а комфортным – 100 Гц и более.

Частота строчной (горизонтальной) развертки определяет, сколько строк формируется на экране монитора в течение одной секунды. Значение этого параметра указывается в килогерцах (КГц). Частота строк напрямую связана с частотой кадров. Так, например, при разрешении 1024 × 768 пикселей и частоте кадровой развертки 85 Гц будем иметь 768 × 85 = 65 280 Гц. С учетом же 10% запаса на синхронизацию, получается значение 71.8 КГц.

Ширина полосы частот видеосигнала измеряется в мегагерцах (МГц) и определяет самые высокие частоты в видеосигнале. Или, иными словами, это количество точек, которое монитор должен отображать в течение секунды. Ширина полосы частот видеосигнала может быть определена как произведение частоты строчной развертки на горизонтальное разрешение. Так, например, при разрешении 1024 × 768 пикселей и частоте строчной развертки 71.8 МГц, с

учетом 10% запаса на синхронизацию, необходимая ширина полосы частот будет равна: $71.8 \times 1024 \times 1.1 \approx 81$ МГц

Основными недостатками мониторов на основе электронно-лучевой трубки являются:

1. высокое потребление электроэнергии и
2. вредное воздействие на здоровье человека.

Наиболее значимые факторы, отрицательно влияющие на здоровье человека, это:

1. электромагнитное, рентгеновское и ультрафиолетовое излучения,
2. неравномерная яркость экрана,
3. нечеткость изображения и
4. выпуклость экрана монитора.

Первым решением, которое хоть как-то ослабляло вредное воздействие мониторов, было применение защитных экранов-фильтров. Они увеличивали контрастность изображения, устраняли блики, а также, в некоторой степени, защищали от ультрафиолетового излучения. Однако этой защиты явно было недостаточно. Поэтому затем стали выпускаться мониторы, соответствующие различным стандартам безопасности.

Первым таким стандартом был шведский стандарт MPR-II. В настоящее время существует еще несколько общепризнанных международных стандартов безопасности – TCO-92, TCO-95 и TCO-99. Они регламентируют такие параметры, как антибликовые свойства покрытия, допустимы величины электромагнитного и рентгеновского излучений, яркость и контрастность изображения, и т.д. Причем, более «новые» стандарты устанавливают все большее количество регламентируемых параметров, а также весе более жесткие значения этих параметров. Так, например, если для мониторов, удовлетворяющих стандарту TCO-92, не требовался защитный фильтр, то стандарт TCO-99 уже гарантирует безопасную работу за таким монитором в течение 8 часов.

Большинство из недостатков, свойственных мониторам на основе ЭЛТ, устранены в мониторах на основе жидких кристаллов. Они применяются в основном в портативных (notebook) компьютерах, хотя имеются модели и в настольном исполнении. Их несомненным достоинством являются то, что они «плоские» и, соответственно, занимают мало места и относительно легкие, а также более экономичны и имеют меньшие излучения. Но, по сравнению с мониторами на основе электронно-лучевой трубки, они имеют более низкие качество воспроизведения изображения, хотя стоимость их несколько выше. В последнее время все же наметилась устойчивая тенденция к снижению их стоимости и улучшению технических характеристик.

Под жидким кристаллом понимается некоторое состояние вещества, в котором оно обладает как свойствами жидкости, так и некоторыми свойствами кристалла. Для изготовления жидкокристаллических экранов используются так называемые *нематические* жидкие кристаллы, молекулы которых имеют форму палочек, или вытянутых пластин.

Жидкокристаллический экран представляет собой два стекла, между которыми находятся молекулы нематических жидких кристаллов слоем 5÷10 мкм. В отсутствие электрического поля такой экран беспрепятственно пропускает свет. Если же к некоторым его точкам приложить электрическое напряжение, то они становятся не прозрачными. Поскольку жидкие кристаллы сами по себе не светятся, то ЖК-мониторам нужна дополнительная подсветка. В качестве источников света для таких экранов используются либо флуоресцентные лампы с холодным катодом, либо электролюминесцентные панели. При этом экран имеет либо заднюю подсветку (backlight), либо боковую (sidelight). Поскольку жидкокристаллические дисплеи используются в основном в портативных компьютерах, то иногда используются экраны без подсветки, которые работают только в отраженном свете (reflective LCD).

Каждой точкой изображения на экране управляет свой электронный переключатель. Информация об изображении подается на экран построчно, а выбор необходимой точки изображения в строке происходит через

соответствующий электронный переключатель. Изменяя амплитуду напряжения управляющего сигнала, можно изменять яркость соответствующих точек. Такая схема управления называется *активной матрицей* и реализуется, как правило, на тонкопленочных полевых транзисторах (TFT – Thin Film Transistor).

В цветных жидкокристаллических мониторах с активной матрицей каждый элемент изображения (пиксель) состоит из трех точек – красной (R), зеленой (G) и синей (B). Соответственно, для каждого цвета точки, составляющей элемент изображения, необходим свой транзистор и фильтр соответствующего цвета. Поэтому реализация таких дисплеев получается не совсем дешевой. Например, для изготовления монитора с разрешением 1024×768 пикселей, помимо всего прочего, необходимо $1024 \times 768 \times 3 = 2\,359\,296$ транзисторов. Поскольку цветные фильтры довольно сильно поглощают свет от подсветки, то ее мощность также должна быть увеличена.

Помимо общих для всех мониторов характеристик, дисплеи на жидких кристаллах имеют свои, специфические, характеристики. Так, для них большое значение имеет такой параметр, как:

1. *время отклика*, характеризующий степень инерционности вывода изображения, и
2. *угол зрения*.

Так, для мониторов с TFT-матрицей средним значениям для времени отклика является 20÷30 мкс, а для угла зрения – $70^0 \div 80^0$.

Сочетание клавиатуры, мыши и монитора в компьютерной системе обеспечивает наиболее современный тип *интерфейса пользователя*, который называется *графическим*. При этом на экране монитора отображаются различные *графические объекты* и *элементы управления*. С помощью мыши и клавиатуры пользователь *изменяет свойства объектов* и *приводит в действие элементы управления*, а на мониторе наблюдает «результаты» этих операций.

Принтер (устройство печати) предназначен вывода информации на бумажный носитель, или получения «твердой копии» (hard copy). По способу получения изображения наиболее распространенными являются:

1. матричные,
2. струйные и
3. лазерные

принтеры.

В матричном («иголочном») принтере изображение формируется из точек, которые получаются в результате ударов цилиндрических стержней («иголок») через красящую ленту по бумаге. При этом качество печати таких принтеров напрямую зависит от количества иголок в печатающей головке. Наиболее распространенными являются 9- и 24-иголчные принтеры. Последние по качеству вывода символьной информации не уступают пишущим машинкам. Основными недостатками струйных принтеров являются невысокая скорость и низкое качество печати. Однако они хорошее зарекомендовали себя при выводе небольших объемов текстовой информации, поскольку являются самыми дешевыми и довольно надежными в эксплуатации печатающими устройствами.

Струйный принтер формирует изображение из точек, образованных при попадании микрокапель красителя («чернил») на бумагу. Выброс микрокапель красителя происходит за счет давления, которое развивается в печатающей головке. Качество печати струйных принтеров зависит от формы микрокапли, а также от характера впитывания красителя бумагой. В этих условиях особую роль играют свойства красителя и качество бумаги. Основным достоинством струйных принтеров является их более низкая стоимость, по сравнению с другими типами, при высоком качестве печати. Однако они требуют более качественного профилактического обслуживания при эпизодическом характере печати. Это обусловлено необходимостью прочистки сопел печатающей головки после более-менее длительного перерыва в работе связанного с засыханием в них «чернил». Наибольшее распространение струйные принтеры

получили в цветной печати, поскольку имеют наилучшее соотношение цена/качество.

Лазерные принтеры обеспечивают наилучшее качество печати, которое в некоторых случаях превосходит полиграфическое. Изображение в лазерных принтерах создается следующим образом:

1. В соответствии с характером поступающей информации лазерная головка, или светодиодная матрица, испускают световые импульсы, которые, попадая на поверхность светочувствительного барабана, вызывают статический заряд в соответствующих участках.
2. При вращении барабана в специальном контейнере – картридже – наполненном красящим составом – тонером – происходит его закрепление на участках, имеющих статический заряд.
3. При дальнейшем проворачивании барабана, происходит его соприкасание с бумагой, в результате чего тонер переносится на бумагу.
4. Закрепление тонера на бумаге осуществляется при ее протягивании через нагревательный элемент, при котором происходит «запекание» полученного изображения.

Основными потребительскими характеристиками принтеров являются следующие:

1. Формат бумаги, на которую осуществляется вывод, например, A4, A3 и т.д.
2. Качество печати, или разрешающая способность, которая измеряется в dpi – dots per inch – точек на дюйм. Для матричных принтеров эта величина имеет значение прядка 120 dpi. Модели среднего класса струйных и лазерных принтеров обеспечивают разрешение до 600 dpi, а высшего – до 2400÷2800 dpi.
3. Производительность, или скорость печати. Для матричных принтеров она оценивается по количеству печатаемых символов в секунду – cps – characters per second, и имеет значение порядка 200÷400 cps. Для струйных и матричных принтеров производительность измеряется в отпечатанных страницах в минуту – ppm – pages per minute, и находится в диапазоне 10÷15 ppm.

4. Удельная стоимость печати одного листа. Она состоит из первоначальной стоимости самого принтера и стоимости расходных материалов. Стоимость же расходных материалов, в свою очередь, состоит из стоимости красящего вещества и стоимости печатающей головки, или светочувствительного барабана, которые постепенно изнашиваются, и подлежат замене. В начальный период эксплуатации наиболее экономичными являются относительно дешевые струйные принтеры. При больших же объемах печати более экономичными становятся лазерные принтеры, поскольку ресурс светочувствительного барабана достаточно высок, и одной заправки картриджа тонером хватает на большое количество отпечатков. Наиболее экономичными, в этом смысле, являются матричные принтеры, тат как стоимость расходных материалов (красящей ленты) у них наименьшая.

Внешние запоминающие устройства (**ВЗУ**) предназначены для длительного хранения и переноса информации между компьютерами. По способу регистрации (записи) информации наиболее распространенными типами ВЗУ являются:

- магнитные, и
- оптические.

Всякое внешнее запоминающее устройство состоит из 2-х частей:

- носителя (диск, лента и т.д.) на котором, собственно, и происходит регистрация информации, и
- накопителя (привод, драйвер, дисковод, «карман» и т.д.), в котором осуществляются операции записи-считывания информации.

Конструктивно накопитель и носитель информации в некоторых случаях могут быть выполнены в виде единого устройства, например, жесткие магнитные диски, или в виде двух отдельных компонент, например, привод лазерного диска и сам лазерный диск.

По способу организации доступа к информации ВЗУ делятся на устройства:

1. последовательного и

2. прямого

доступа. В устройствах последовательного доступа для того, чтобы прочитать (или записать) n-ую порцию данных необходимо предварительно прочитать (или записать) n-1 таких блоков. В устройствах прямого доступа запись (или считывание) требуемого блока данных выполняется непосредственно. Всякое устройство прямого доступа всегда можно использовать и в последовательном режиме, но обратное – никогда. Наиболее распространенными устройствами последовательного доступа являются магнитные ленты, а прямого – диски (магнитные и оптические).

Наиболее важными характеристиками любого внешнего запоминающего устройства, сточки зрения пользователя, являются следующие три:

1. емкость,
2. скорость записи-чтения информации, и
3. удельная стоимость хранения единицы данных, которая оценивается как отношение стоимости накопителя и носителя информации к их емкости.

По человеческим меркам устройства ввода-вывода работают очень быстро, однако, по сравнению со скоростью обмена информацией между другими устройствами компьютера, их скорость крайне низкая.

Магнитные диски обеспечивают память большой емкости, позволяют подолгу и без значительных затрат хранить большие объемы информации, будь то программы или данные. Если оперативную память компьютера можно сравнить с памятью человека, то магнитные диски (или внешние запоминающие устройства) сопоставимы с библиотекой. Человеку доступна хранящаяся в библиотеке информация, но прежде, чем ею воспользоваться, он должен ее прочесть. Точно так же информация, хранящаяся во внешней памяти, должна быть перенесена в оперативную память, чтобы с ней мог работать центральный процессор. Основными характеристиками магнитных дисков являются емкость и скорость записи-чтения информации.

В современном компьютера наиболее часто применяются два типа дисков: гибкие (дискета, floppy disk) и жесткие («винчестер», hard disk).

Гибкий диск представляет собой пластиковый круг, с обеих сторон покрытый магнитным слоем (как магнитофонная лента), и помещенный в бумажный или пластиковый несъемный конверт. В конверте есть специальные отверстия, через которые осуществляется вращение диска и подвод к нему головок записи-чтения. Непосредственная запись и чтение информация с магнитного диска осуществляется в специальном устройстве, которое называется дисковод (драйвер, «карман», и т.д.).

Жесткий диск – это герметически запаянный металлический корпус, в который помещены головки записи-чтения и несколько, посаженных на одну ось, дисков. Связь с остальными электронными устройствами компьютера осуществляется через специальный разъем, расположенный на его корпусе. За счет такой конструкции емкость и скорость записи-чтения жесткого диска в несколько сот раз больше, чем гибкого диска. Наиболее распространенными форматами как жестких, так и гибких дисков, которые определяются диаметром их рабочих пластин, являются 5.25” и 3.5”.

Можно спорить о том, находится ли информация, хранящаяся на магнитных дисках, «внутри» компьютера или нет. Если, например, информация записана на гибкий диск, то его можно вынуть из дисковода и полностью отделить от компьютера. Таким образом, магнитные диски могут рассматриваться не как память внутри компьютера, а как устройства ввода-вывода. По сравнению с процессором и оперативной памятью они работают довольно медленно, но все же намного быстрее тех типов устройств ввода-вывода, которые упоминались выше; в этом их преимущество перед последними.

7. Физическая организация хранения информации на магнитных дисках

В виду того, что основная масса информации, обрабатываемой на компьютере, хранится на магнитных дисках, и большинство программ в той или иной степени взаимодействуют с ними, далее несколько подробнее

остановимся на способах организации физического хранения и обмена информацией на магнитных дисках.

Запись информации на магнитные диски выполняется с помощью головок записи-считывания (read-write heads) по концентрическим окружностям, которые называются дорожками, или треками (tracks). Каждая дорожка имеет свой уникальный номер, начиная с 0 у края диска. Дорожки разделены на более мелкие единицы хранения информации, которые называются секторами (sectors). Размер сектора фиксирован, и для большинства дисков составляет 512 байт. Сектор является слишком маленькой единицей хранения и обмена информацией, особенно для больших дисков. Поэтому в качестве компромисса между дорожкой и сектором был принят кластер (cluster) – несколько подряд идущих секторов. Обмен информацией между оперативной памятью компьютера и диском выполняется блоками данных, равным кластеру. Дисковое пространство под файлы так же выделяется в кластерах. При этом последний выделенный файлу кластер может быть заполнен лишь частично. Размер кластера – величина, фиксированная для каждого диска, и может быть установлена в процессе подготовки его к эксплуатации, который называется форматирование (formatting). Выбор размера кластера – это всегда компромиссное решение между скоростью записи-считывания и эффективностью использования дискового пространства. С одной стороны он может быть равен сектору, и дисковое пространство при этом будет использовано максимально эффективно, а скорость обмена – минимальной. С другой стороны его размер ограничен размером дорожки; скорость обмена при этом максимальна, а эффективность использования – минимальна. Поверхности (sides) магнитных дисков так же пронумерованы. Нумерация начинается с 0 для самой верхней поверхности, самого верхнего (или единственного) диска. Каждому номеру поверхности однозначно соответствует такой же номер головки записи-чтения. Конструкция приводов магнитных дисков такова, что головки записи-чтения не могут позиционироваться на необходимые дорожки каждая индивидуально – они перемещаются все вместе. Каждое такое их

положение над определенной дорожкой называется цилиндром (cylinder). Если мысленно соединить по вертикали все дорожки с одинаковыми номерами на всех поверхностях магнитных дисков, то получится цилиндр. Аппаратные и программные средства компьютера часто работают, используя цилиндры. Когда данные записываются на диск по цилиндрам, к ним можно обращаться без необходимости перемещения головок записи-чтения. Поскольку перемещение головок медленно, по сравнению со скоростью вращения пластин и переключениями между отдельными головками, то использование цилиндров значительно уменьшают время доступа к данным.

В некоторых случаях один физический диск может быть разделен на несколько логических дисков. Это делается в случае, когда размер физического магнитного диска больше, чем может поддерживать конкретная операционная система. Физический диск разделяется на логические так же в случае, если на одном компьютер используется несколько операционных систем, которые поддерживают несовместимые файловые системы.

Как было отмечено выше, дисковое пространство под файл выделяется в блоках фиксированного размера – кластерах. Кластеры, занимаемы файлом, не всегда выделяются подряд – довольно часто файлы бывают фрагментированными. Информация об именах файлов, занимаемых ими кластерах и т.д. хранится в специально отведенном для этого месте. Но это уже относится не к физической, а к логической организации хранения информации на магнитных дисках, которая более подробно изложена в лекции, посвященной операционной система.

Лазерные (оптические) диски. В последнее время широкое распространение получили ***CD-ROM*** (Compact Disk Read Only Memory) или «*лазерные диски*». Информация на пластмассовую основу диаметром 5.25” записывается с помощью лазерного луча. Как видно из аббревиатуры ***CD-ROM***, лазерный диск используется только для чтения информации. Это связано с технологией записи информации – повторная запись информации на один и тот же диск (носитель) невозможна. ***CD-ROM*** используются в основном для

распространения больших объемов информации (программное обеспечение, музыка, электронные энциклопедии, фильмы и т.д.). Столь широкое использование однократно записываемых носителей информации обусловлено их низкой удельной стоимостью хранения информации. Она гораздо ниже по сравнению с другими *внешними запоминающими устройствами*, такими, например, как магнитные диски.

8. Уровни памяти компьютера

Из приведенного выше обзора памяти компьютера следует, что она разделена на три уровня, в зависимости от емкости, скорости доступа и цены. Это:

- внешняя память или внешние запоминающие устройства; наиболее емкая и наиболее медленная; информация в ней может храниться сколь угодно долго, даже после выключения питания.
- оперативная память или оперативное запоминающее устройство предназначено для хранения программ и данных, необходимых при их выполнении; занимает промежуточное положение по емкости и скорости доступа; информация «стирается» после выключения питания.
- сверхоперативная память – регистры центрального процессора; емкость регистра общего назначения соответствует длине машинного слова, их количество ограничено, а скорость обмена информацией – максимальная; информация в них так же «стирается» после снятия питания.

Термин «компьютер» довольно часто используется и для обозначения аппаратного обеспечение, и для обозначения всей компьютерной или вычислительной системы, т.е. и программного и аппаратного обеспечения. Путаницы с использованием данного термина, как правило, не происходит – его значение всегда можно определить из контекста, в котором он используется.

ЛЕКЦИЯ №2

ОБСЛУЖИВАНИЕ ОПЕРАЦИОННОЙ СИСТЕМЫ MICROSOFT WINDOWS

План

1. Состав и назначение программного обеспечения компьютера
2. Структура компьютерной программы
3. Графический интерфейс пользователя
4. Типы интерфейсов приложений Microsoft Windows
5. Файловые системы
6. Получение справочной информации

1. Состав и назначение программного обеспечения компьютера

Секрет столь широкого использования компьютеров в различных областях человеческой деятельности кроется в программном обеспечении, которое позволяет использовать одну и ту же машину, построенную из кремния и стали, для решения огромного множества самых разнообразных задач. В отличие от первых программ, вводившихся в машину посредством утомительного переключения множества тумблеров и цифро-наборных устройств, программное обеспечение, вдохнувшее жизнь в современный компьютер, записывается, как правило, на магнитных дисках и вступает в работу сразу после включения машины. Однако по существу команды компьютера, хотя они и задаются теперь гораздо более удобным способом, заметных изменений не претерпели. Любой компьютер должен разложить задание на последовательность машинных команд, а затем выполнять их одну за другой.

В английском языке для программного обеспечения выбрано (а точнее, создано) довольно удачное слово *software* (буквально – «мягкое изделие»), которое подчеркивает равнозначность программного и аппаратного обеспечения («железки» – *hardware* – «жесткое изделие»), и вместе с тем говорит о его гибкости, способности модифицироваться, приспосабливаться,

развиваться. Введение этих терминов было связанно с необходимостью провести четкую грань между командами-инструкциями, управляющими компьютером, и его физическими компонентами или аппаратным обеспечением, которое, собственно, и составляет компьютер. Поэтому переключение компьютера с построения рисунка на разработку стандартного контракта или с разработки архитектурного проекта на создание карты погоды земного шара осуществляется простым изменением последовательности команд, управляющих его работой, т.е. программ.

Прежде, чем обсуждать состава и назначения программного обеспечения, введем еще два термина: *программист* и *пользователь*. Если программа сложная, в ее создание вовлекается большая группа людей, выполняющих различные функции, то мы будем употреблять общий термин «программист» для обозначения человека, который проектирует, записывает и совершенствует новую программу, или модифицирует старую. Человек, для которого пишется программа и который, вероятно, будет снабжать ее входными данными и использовать полученные результаты, называется пользователем. И хотя один и тот же человек может быть и пользователем, и программистом, важно различать эти две ипостаси.

Общий термин программное обеспечение (*software*) применяется для обозначения программ, используемых в компьютерной системе. Эти программы подразделяются на два больших класса:

1. прикладное программное обеспечение (ППО) и
2. базовое программное обеспечение (БПО).

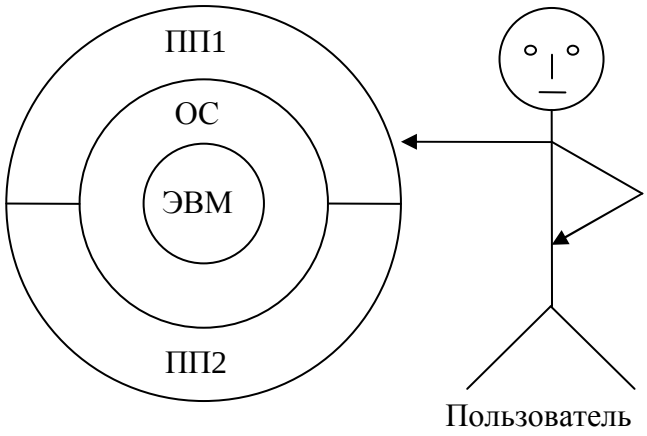
Прикладное программное обеспечение образуют программы для пользователей и осуществляющие информационные процессы, которые нужны пользователям.

Центральное место в базовом программном обеспечении занимает специальная совокупность программ, называемая *операционной системой*, которая обеспечивает взаимодействие:

1. человека,

2. компьютера и
3. программ.

Такое взаимодействие выполняется в соответствии с Рис. 2.1.



ЭВМ – электронная вычислительная машина (компьютер)
ОС – операционная система
ПП1, **ПП2** – прикладная программа1, прикладная программа2

Рис. 2.1. Взаимодействие пользователя, компьютера и программ

Операционная система дает указание компьютеру, как интерпретировать команды и данные, как распределять аппаратные ресурсы для выполнения заданий и как управлять периферийными устройствами (например, принтером или дисплеем). Она также обеспечивает возможность непосредственного взаимодействия человека и компьютера, выполняя такие действия, как хранение программ и данных на внешних запоминающих устройствах.

Если рассматривать операционную систему как «режиссера» компьютерного действа, то прикладные программы играют роль «артистов». Именно благодаря таким программам, как текстовые процессоры, игры и электронные таблицы, компьютер приобретает удивительную разносторонность.

Понятие «базовое программное обеспечение» включает также такие программы, как трансляторы и утилиты. Транслятор – это программа, которая принимают в качестве исходных данных другую программу, написанную на так называемом языке программирования высокого уровня, отдаленно

напоминающем язык человека, и отличном от машинного кода, с которым имеет дело процессор, и переводит эту программу в машинный язык, представляющий собой комбинацию нулей и единиц, которые компьютер обрабатывает как последовательности электрических импульсов. Утилиты выполняют рутинные, но часто крайне необходимые, функции, например, удаление ненужной информации с магнитных дисков. Эти скромные «рабочие лошадки» помогают решать типовые задачи обработки информации.

И так, полная *вычислительная система*, включающая аппаратуру (*hardware*) и программное обеспечение (*software*), создает среду, в которой могут разрабатываться, храниться и выполняться программы.

2. Структура компьютерной программы

Любая компьютерная программа состоит из 3-х частей (Рис. 2.2):

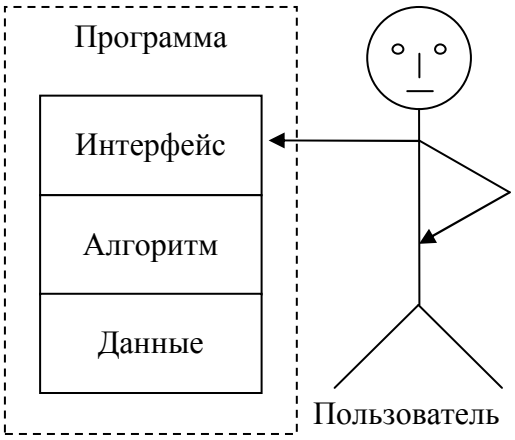


Рис. 2.2. Структура компьютерной программы

1. *Интерфейс* – набор правил взаимодействия между пользователем и программой;
2. *Алгоритм*, или *бизнес-логика* – набор правил преобразования информации из входной в выходную;
3. *Данные* – то, что обрабатывает программа.

Все три части обязательно присутствуют в любой компьютерной программе. Однако, «удельный вес» каждой из них зависит от назначения программы. Так, например, в научно-технических (инженерных) программах

преобладают алгоритмы, а интерфейс и данные у них, как правило, не сложные. Экономические задачи, наоборот, обрабатывают большие объемы информации (т.е. преобладают данные), а алгоритмы и интерфейс у таких программ относительно простые. Игровые программы обладают хорошо проработанными интерфейсами и захватывающими алгоритмами, и несложными данными.

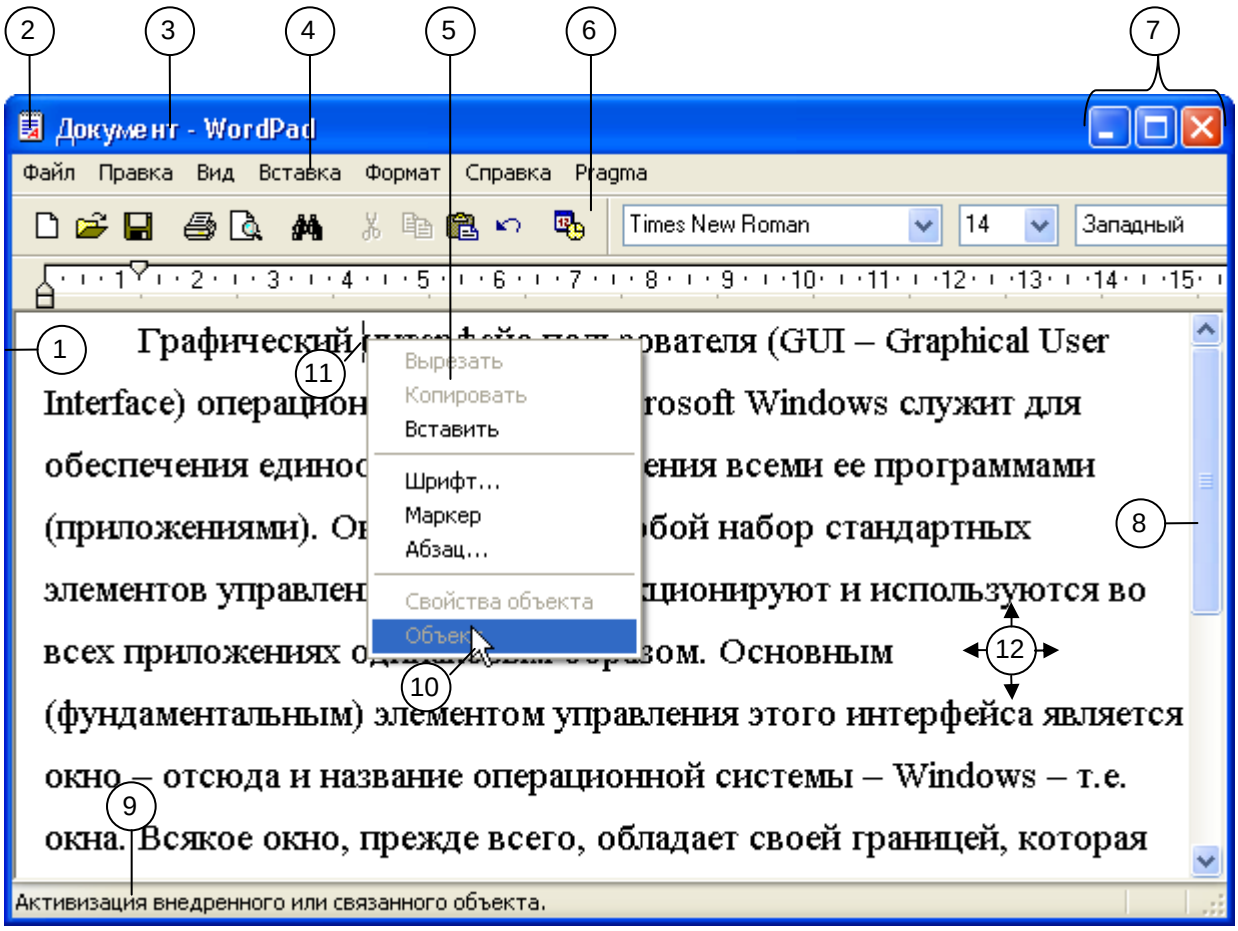
3. Графический интерфейс пользователя

Графический интерфейс пользователя (GUI – Graphical User Interface) операционной системы Microsoft Windows служит для обеспечения единообразного управления всеми ее программами (приложениями). Он представляет собой набор стандартных элементов управления, которые функционируют и используются во всех приложениях одинаковым образом. Основным (фундаментальным) элементом управления этого интерфейса является окно – отсюда и название операционной системы – Windows – т.е. окна. Всякое окно, прежде всего, обладает рамкой, которая является его границей, и в некоторых случаях может отображаться явно, а в других – неявно, но присутствует – всегда. Содержание же окна зависит от его назначения – это может быть и элементарная надпись, или кнопка, и окно многодокументного приложения, которое само может содержать другие (дочерние) окна. Структура типового окна приложения Windows представлена на Рис. 2.3.

Оно состоит из таких элементов, которые выполняют следующие функции:

Рамка окна (поз. 1 Рис. 2.3) служит для указания границ текущего окна. При наведении указателя мыши на границу окна он принимает вид двунаправленной стрелки (↔). В этом положении при нажатой левой кнопке мыши можно изменять размеры окна.

⚠ *Не всякое окно может менять свои размеры!*



1. Рамка окна
2. Кнопка вызова системного меню
3. Заголовок окна приложения
4. Основное меню приложения
5. Контекстное меню
6. Панель инструментов
7. Кнопки управления окном
8. Полоса прокрутки
9. Строка состояния
10. Указатель мыши
11. Курсор
12. Рабочая (клиентская) область окна

Рис. 2.3. Типичное окно приложения
операционной системы Microsoft Windows

Кнопка вызова системного меню (поз. 2 Рис. 2.3), которое служит для изменения положения и размеров окна, а также завершения работы приложения. Это старый и редко используемый элемент управления – гораздо удобнее для этих целей воспользоваться кнопками в правом верхнем углу окна (поз. 7 Рис. 2.3).

Заголовок окна приложения (поз. 3 Рис. 2.3). В нем отображается название приложения и документа, который это приложение обрабатывает. С помощью заголовка можно перемещать окно по экрану монитора.

Основное меню приложения (поз. 4 Рис. 2.3). Самый старый элемент управления. Оно обеспечивает доступ к абсолютному большинству

возможностей приложения. При этом все они сгруппированы в виде древовидной структуры. Например, в пункте меню **Файл** собраны все функции приложения связанные с открытием, закрытием и сохранением документов, а в пункт **Правка** – содержит команды редактирования открытого документа. Работа с меню имеет некоторые особенности:

а) Пункт меню, который отображается бледным цветом, является недоступным в текущем состоянии.

б) Пункт меню, который заканчивается стрелкой (▸), имеет вложенные команды нижнего уровня, и при наведении на него указателя мыши (или щелчка по нему) они становятся доступными.

в) Если пункт меню заканчивается многоточием (...), то при его выборе раскрывается окно диалога, в котором можно уточнить параметры выполнения команды. Например, при выполнении команды **Файл** ⇨ **Открыть**, в раскрывшемся окне диалога необходимо уточнить имя файла, и место его расположения.

г) Выбор пункта меню, который никак не выделен, вызывает немедленное выполнение указанной команды.

Контекстное меню (поз. 5 Рис. 2.3) – самый удобный элемент управления. Активизируется щелчком правой кнопки мыши. Состав его пунктов зависит от:

1. объекта, по которому выполнен щелчок, и
2. состояния этого объекта.

Т.е., в нем собраны допустимые действия над выделенным объектом в текущем его состоянии.

Панель инструментов (поз. 6 Рис. 2.3) предназначена для группировки только тех элементов управления, которые нужны для выполнения определенного вида работы. Наиболее часто в приложениях используются две панели инструментов:

1. **Стандартная** и
2. **Форматирование**.

Стандартная панель инструментов содержит элементы управления, которые наиболее часто используются, т.е. при открытии и сохранении документа, его редактировании и т.д. Панель инструментов **Форматирование** содержит элементы управления, которые применяются для изменения внешнего вида документа.

Три кнопки в правом верхнем углу окна (поз. 7 Рис. 2.3) служат для сворачивания (), разворачивания () и закрытия приложения (), соответственно.

Полоса прокрутки (поз. 8 Рис. 2.3) служит для «пролистывания» документа, который полностью не помещается в окне. Бывают вертикальные и горизонтальные полосы прокрутки.

Строка состояния (поз. 9 Рис. 2.3) предназначена для отображения текущего состояния приложения.

Указатель мыши (поз. 10 Рис. 2.3) перемещается синхронно с мышью. Его форма зависит от:

1. объекта, на который он указывает и
2. состояния этого объекта.

При перемещении в пределах текста он принимает вид вертикальной черточки с засечками (I). Щелчок левой кнопкой мыши при этом вызывает появление курсора (поз. 11 Рис. 2.3) – вертикальной мигающей черточки (|), которая показывает позицию редактирования в текстовом документе.

Всю оставшуюся площадь окна приложения (поз. 12 Рис. 2.3) занимает рабочая (клиентская) область, в которой отображается текущее состояние редактируемого документа.

4. Типы интерфейсов приложений Microsoft Windows

Всякое приложение Microsoft Windows может использовать один из следующих стандартных интерфейсов:

1. *Интерфейс командной строки* служит в основном для поддержки выполнения старых приложений MS-DOS и активизируется по команде **Пуск ⇨ Выполнить**.

2. *Однодокументный интерфейс* (SDI – Single Document Interface). Приложения с таким типом интерфейса одновременно могут работать только с одним документом. Для редактирования следующего документа необходимо запустить на выполнение еще одну копию этого приложения. Этот тип интерфейса имеют, например, такие приложения как Проводник, Блокнот, Paint, WordPad.
3. *Многодокументный интерфейс* (MDI – Multiple Document Interface), например, Microsoft Word или Microsoft Excel. Приложения этого типа одновременно могут открывать для редактирования несколько документов. Характерной особенностью этого типа приложений является наличие пункта основного меню **Окно**, который служит для управления дочерними окнами одновременно редактируемых документов.
4. *Интерфейс окна диалога*, например, Калькулятор, Таблица символов. Диалоговые окна являются перемещаемыми, но не меняют своих размеров и не могут быть развернуты на весь экран. Специальным типом окна диалога является **Мастер**. Он «руководит» последовательностью действий пользователя при ответах на множество задаваемых вопросы, и обязательно содержит кнопки **Далее**, **Назад**, **Отмена** и **Готово**.

   Окна диалога, которые используются как вспомогательные элементы управления основных приложений Windows, бывают двух видов:

   а) *модальные* – которые приостанавливают выполнение приложения, из которого они были вызваны, до своего закрытия (например, окно **Параметры** в большинстве приложений Microsoft Windows) и

   б) *немодальные* – которые не приостанавливают выполнение «своего» приложения до их закрытия (например, окно **Найти и заменить** в Microsoft Word).

Все окна операционной системы Microsoft Windows организованы в виде некоторой древовидной (иерархической) структуры, которая называется Z-order (Z-порядок). То есть, всякое окно является «подокном» окна более высокого уровня иерархии (родительского), а так же может иметь «подчиненные»

(дочерние) окна. На самом верху этой иерархии находится окно, которое называется **Рабочий стол** – он родительского окна не имеет.

5. Файловые системы

Одной из основных функций операционной системы является организация хранения информации на внешних запоминающих устройствах (ВЗУ) компьютера. Основу хранения информации на ВЗУ (магнитных и оптических дисках, flash-дисках и т.д.) составляет файловая система. Под файловой системой понимают:

1. набор правил хранения информации на ВЗУ, а так же
2. набор программ, которые эти правила реализуют.

Операционная система Microsoft Windows XP поддерживает следующие файловые системы:

1. **NTFS** – основная файловая система Windows XP. Она отличается повышенной надежностью и разграничением доступа к информации.
2. **FAT** – файловая система предыдущих версий операционной системы Windows XP. Оставлена для обеспечения совместимости с ними.
3. **CDFS** – файловая система оптических дисков.

Основными объектами любой файловой системы являются:

1. файл,
2. папка и
3. диск.

Файлом называется поименованная совокупность однотипных данных на ВЗУ. Он обладает множеством различных параметров. Основные из них это:

1. имя,
2. тип и
3. размер.

Имя файла указывается пользователем при его создании. При этом необходимо придерживаться нескольких правил:

- имя файла должно быть уникальным в пределах папки;

- длина имени файла не должна превышать 255 символов;
- имя файла не должно совпадать с одним из зарезервированных имен, таких как, **COM**, **LPT** и т.д.
- в имени файла не должны встречаться некоторые символы, такие, например, как «*****», «**/**», «**?**» и т.д.

На тип файла указывает *расширение имени файла*. Расширение имени файла – это несколько символов (как правило, три) после последней справа точки в имени файла. Тип файла указывает способ кодирования информации в нем, а также программу, в которой он был создан. В операционной системе Windows существует несколько зарезервированных типов файлов. Так, расширение **.TXT** имеют файлы, созданные в программе Блокнот, **.DOC** – в Microsoft Word, а **.XLS** – Microsoft Excel. Расширение имени файла является необязательной характеристикой файла. При его отсутствии невозможно определить программу, предназначенную для работы с ним.

Специальным типом файлов является *ярлыки*. Они предназначены для обеспечения быстрого доступа к другим объектам файловой системы. То есть, ярлык – это файл, который содержит «адрес» другого объекта файловой системы и имеет расширение **.LNK**.

Папки – это контейнерные объекты, которые могут содержать файлы и другие папки. Они предназначены для группировки файлов в виде древовидной структуры. Имена папок строятся по тем же правилам, что и файлов, только без расширения.

Диски бывают 2-х видов:

- физические и
- логические.

Физические диски – это реальные ВЗУ компьютера, а логические – это части физических. То есть, физический диск может быть разбит на несколько логических дисков, каждый из которых может иметь свою файловую систему. Диски именуются начальными буквами латинского алфавита, за которыми обязательно идет двоеточие «**:**», например, «**C:**».

Для работы с объектами файловой системы (дисками, папками и файлами) предназначены специальные программы, которые называются *файловыми менеджерами*. В операционной системе Windows такой файловый менеджер называется *Проводник*.

6. Получение справочной информации

Каждый пользователь операционной системы Microsoft Windows рано или поздно сталкивается с вопросом: «Как выполнить то, или иное действие?» При этом у новичков такие вопросы возникают более часто, у опытных пользователей – реже. Разработчики операционной системы снабдили ее обширной и мощной справочной системой. Она обеспечивает пользователя поддержкой практически в любой ситуации. Большинство приложений Microsoft Windows так же могут вывести на экран справку, или организовать помощь в форме диалога.

В Microsoft Windows доступны различные виды справки:

- В режиме командной строки для MS-DOS-команд.
  а) По команде **HELP** будет выдан весь список команд, доступных в режиме MS-DOS.
  б) По команде **HELP <имя команды>** будет представлен синтаксис данной команды и обзор допустимых параметров.
- Контекстная помощь по элементам управления окон диалога может быть получена двумя щелчками мыши, двумя способами.

  Первый способ.
  а) Щелкнуть по кнопке со знаком вопрос () в правом верхнем углу окна диалога. Указатель мыши при этом дополнится знаком вопроса () (Рис. 2.4).
  б) Щелкнуть по элементу окна, о котором необходимо получить справку. В результате рядом с выбранным элементом появится текст справки.

Рис. 2.4. Получение справки в окне диалога (первый способ)

Второй способ (прямая помощь).

1. Щелкнуть правой кнопкой мыши по интересующему элементу управления. При этом появится кнопка с вопросом «Что это такое?» (Рис. 2.5).
2. Щелкнуть по кнопке с вопросом «Что это такое?» и на экране появится справка о выбранном элементе управления.

- Справочная система из **Главного меню**.

Справочную систему операционной системы Microsoft Windows можно вызвать 2-я способами:

1. по команде **Пуск ⇨ Справка и поддержка** или
2. нажатием клавиши **F1** (при этом, если активно какое-либо приложение, то вызывается справка данного приложения).

В результате раскрывается окно **Центр справки и поддержки**, которое состоит из 4-х вкладок: **Содержание**, **Указатель**, **Избранное** и **Журнал** (Рис. 2.6.).

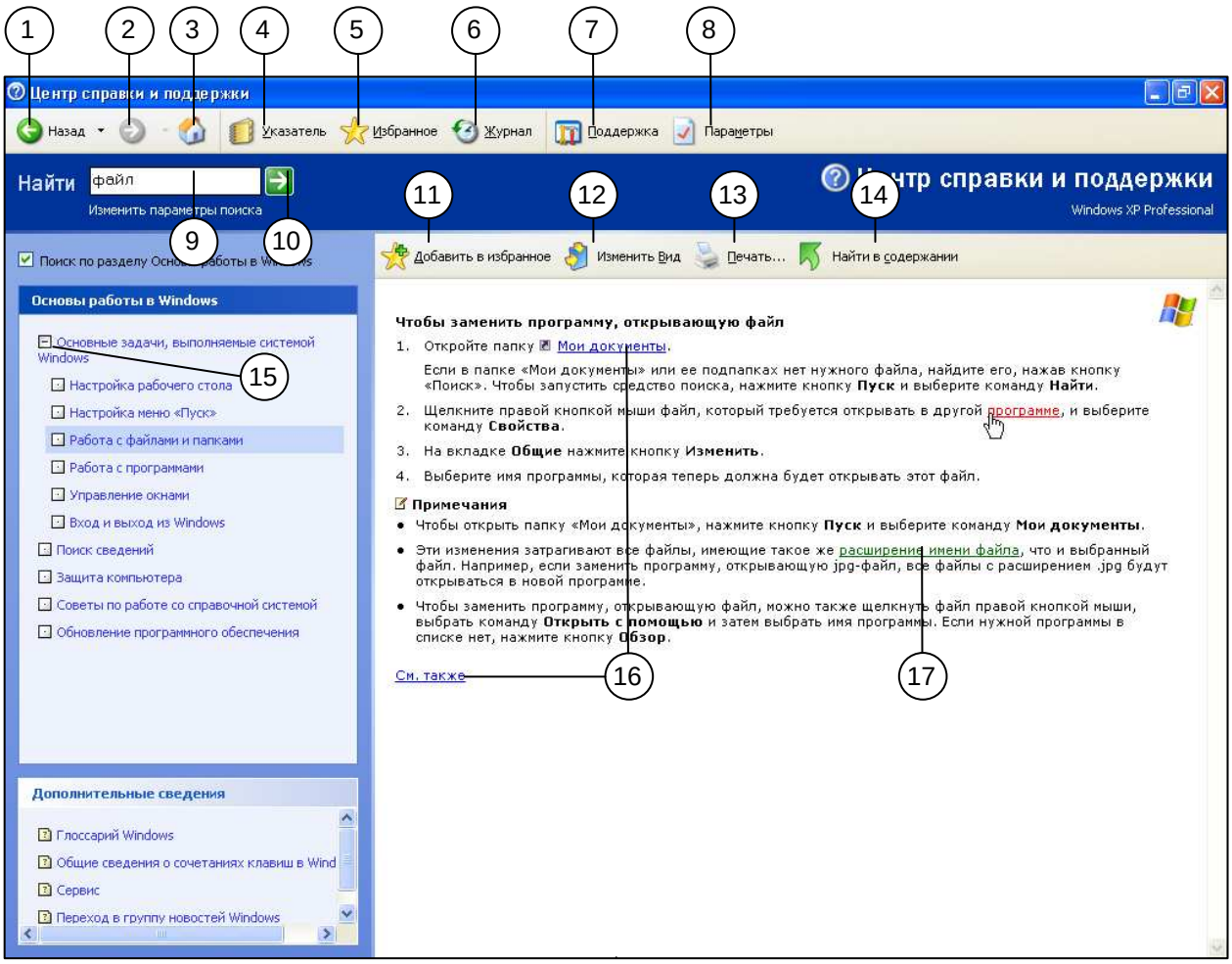
Рис. 2.5. Получение справки в окне диалога (второй способ)

Управление окном справочной системы Microsoft Windows **Центр справки и поддержки** выполняется при помощи элементов управления следующим образом:

Кнопка **Назад** (поз. 1. Рис. 2.6) позволяет возвращаться к предыдущему тексту справки.

Кнопка **Вперед** (поз. 2. Рис. 2.6) позволяет перемещаться вперед по цепочке просмотренных справок. Она становится доступной только после выполнения перехода назад.

Кнопка **Домой** (поз. 3. Рис. 2.6) служит для возврата к начальной странице справочной системы, которая содержит темы, организованной в виде древовидной структуры.



1. Кнопка **Назад**
2. Кнопка **Вперед**
3. Кнопка **Домой**
4. Кнопка **Указатель**
5. Кнопка **Избранное**
6. Кнопка **Журнал**
7. Кнопка **Поддержка**
8. Кнопка **Параметры**
9. Поле редактирования **Найти**
10. Кнопка **Начать поиск**
11. Кнопка **Добавить в избранное**
12. Кнопка **Изменить вид**
13. Кнопка **Печать**
14. Кнопка **Найти в содержании**
15. Кнопка сворачивания/разворачивания
16. Гиперссылка
17. Определение термина

Рис. 2.6. Окно справочной системы Microsoft Windows

Кнопки **Указатель** (поз. 4. Рис. 2.6), **Избранное** (поз. 5. Рис. 2.6) и **Журнал** (поз. 6. Рис. 2.6) служат для перехода к соответствующим вкладкам справочной системы.

Кнопка **Поддержка** (поз. 7. Рис. 2.6) служит для получения доступа к Wed-узлу Microsoft и просмотра там самых свежих сведений об операционной системе, для обращения к другим ресурсам Интернета, или запроса помощи коллег.

Кнопка **Параметры** (поз. 8. Рис. 2.6) используется для настройки параметров **Центр справки и поддержки**.

Поле редактирования **Найти** (поз. 9. Рис. 2.6) служит для ввода ключевых слов поиска.

Кнопка **Начать поиск** (поз. 10. Рис. 2.6) позволяет начать поиск термина, введенного в поле редактирования **Найти**.

Кнопка **Добавить в избранной** (поз. 11. Рис. 2.6) используется для включения просматриваемой страницы справочной системы в список избранных страниц, отображаемых на соответствующей вкладке.

Кнопка **Изменить вид** (поз. 12. Рис. 2.6) позволяет изменить вид окна справки.

Кнопка **Печать** (поз. 13. Рис. 2.6) позволяет распечатать текст справки. После указания объекта печати открывается диалоговое окно **Печать**, в котором можно задать параметры печати. При этом можно распечатать все подчиненные справки данной темы.

Кнопка **Найти в содержании** (поз. 14. Рис. 2.6) выполняет переход на вкладку **Содержание**, причем в левой ее части отображается тот участок иерархии справочных тем, к которой относится данная тема.

Для ускорения перемещения по справочной системе в тексте справки могут располагаться различные кнопки и гиперссылки. Так, если фрагмент текста отображается голубым цветом с подчеркиванием (поз. 16. Рис. 2.6), то после щелчка по нему выполняется переход к адресуемой теме. Если же слово в тексте отображается зеленым цветом с подчеркиванием (поз. 17. Рис. 2.6), то оно представляет собой определение термина, и после щелчка на нем выводится его краткое описание. Маленькие кнопки слева от заголовков в тексте справки (поз. 15. Рис. 2.6) позволяют сворачивать/разворачивать подчиненный данному заголовку текст. В конце текста помощи может располагаться элемент **См. также** (поз. 16. Рис. 2.6), щелкнув на котором, можно выбрать тему, имеющую отношение к текущей.

Вкладка **Содержание** открывается по умолчанию и обеспечивает доступ к справочной системе, организованной в виде древовидной структуры иерархии тем – т.е., как в любом справочнике – по оглавлению. Выделенная тема раскрывается щелчком мыши на ее имени, которое представляет собой гиперссылку на соответствующий фрагмент справочного текста. После этого в правой части окна отображается содержание тематической группы, которое, как правило, включает ряд других тематических групп, оформленных в виде гиперссылок. При необходимости можно открыть одну из тематических групп только что отрытой. Щелчок мышью по последней в иерархии гиперссылке приводит к отображению необходимой темы справочной системы.

Вкладка **Указатель** позволяет вести поиск по ключевым словам справочной системы. После ввода начальных букв ключевого слова в поле редактирования списка отображаются соответствующие разделы справочной системы. Для просмотра темы необходимо выполнить два щелчка:

1) один раз по выбранной теме, и
2) второй раз – по кнопке **Показать**, или

- двойной щелчок по выбранной теме в списке.

Разумеется, что не каждое слово является ключевым и снабжено справкой.

Вкладка **Журнал** позволяет обратиться к уже просмотренным ранее страницам справочной системы.

Вкладка **Избранное** дает возможность составить список часто используемых тем справочной системы для быстрого доступа к ним. Работа с часто используемыми темами состоит из 2-х этапов. На первом этапе просматриваемая тема добавляется в список избранных щелчком по кнопке **Добавить в избранной** (поз. 11. Рис. 2.6) в окне **Центр справки и поддержки**. На втором этапе необходимо перейти на вкладку **Избранное** и выбрать нужную тему в списке избранных.

Самый последний, и самый простой, способ получения необходимой справочной информации – посредством поиска по ключевому слову в базе данных справочной системы. Для этого необходимо просто ввести ключевое

слово (или несколько слов, разделенных запятыми) в поле редактирования **Найти** (поз. 9. Рис. 2.6) и щелкнуть по кнопке **Начать поиск** (поз. 10. Рис. 2.6), или нажать клавишу **Enter**. После этого в списке **Результаты поиска** будут отображаться наименования соответствующих тем справки. Они сгруппированы в 3 раздела:

1. темы, в заголовках которых встретился указанный термин;
2. темы, в тексте справки которых встретился указанный термин;
3. темы из информационной базы Microsoft.

Для просмотра требуемой темы необходимо просто щелкнуть по ней в нужном списке. Слишком долго затянувшийся поиск можно остановить щелчком по кнопке **Стоп**. Параметры поиска можно изменить, щелкнув по гиперссылке **Изменить параметры поиска**, которая находится под полем редактирования **Найти**.

ЛЕКЦИЯ №3

ОСНОВЫ ОБРАБОТКИ ТЕКСТОВОЙ ИНФОРМАЦИИ

План

1. Кодирование символьной информации и структура текстового файла
2. Классификация программных средств обработки текстовой информации и текстовые документы
3. Запуск Microsoft Word
4. Окно Microsoft Word
5. Ввод текста
6. Перемещение по документу
7. Выделение текста
8. Редактирование текста
9. Отмена и возврат действий
10. Сохранение документа
11. Завершение работы

1. Кодирование символьной информации и структура текстового файла

Информация различного вида – текстовая, звуковая, графическая и т.д. – в компьютере представлена в файлах различного *типа*, или *формата*. Для хранения символьной информации, такой как письма, записки, справки, объявления, отчеты, статьи, исходные тексты программ и многое другое, используются так называемые *текстовые файлы*. Способ кодирования информации в текстовом файле достаточно прост – в нем каждому символу информации поставлено в соответствие некоторое целое положительное число. Наиболее часто символьная информация кодируется с помощью восьмиразрядного двоичного кода. С помощью 8-ми двоичных разрядов можно закодировать $2^8 = 256$ различных символов. Этого вполне достаточно, чтобы представить все символы английского и русского языков, как строчные, так и прописные, а также все знаки препинания, цифры, знаки арифметических

операций и некоторые другие общепринятые специальные символы, например «§». Для стандартизации кодирования символьной информации применяются так называемые *кодовые таблицы символов*. Первые 128 кодов во всех таблицах одинаковы. Здесь расположены управляющие символы, символы английского алфавита, знаки препинания, цифры и знаки арифметических операций. Во второй половине кодовой таблицы (коды с номерами 128 ÷ 255) расположены символы национальных алфавитов. Для стандартизации расположения символов национальных алфавитов используются *кодовые страницы* (Code Page – CP). Символы русского алфавита в операционной системе MS-DOS кодируются с помощью кодовой страницы 866 (CP-866), а в Windows – 1251 (CP-1251). Существование двух различных кодовых таблиц для кодирования символов русского языка вызывает определенные, хотя и преодолимые, но все-таки трудности при использовании одного и того же файла в различных операционных системах.

Текстовый файл – простейший из всех известных способов организации данных. Вся информация в нем представлена в виде некоторой последовательности байт, которые без всяких преобразований можно вводить с клавиатуры и посылать на дисплей или принтер. Имя текстового файла, как правило, имеет расширение **.TXT**.

Подобно любому документу на бумаге, текстовый файл состоит из набора строк переменной длины. Каждая такая строка заканчивается двумя управляющими символами: «Возврат каретки» (код 13) и «Новая строка» (код 10). В последнем байте текстового файла иногда записывается управляющий символ «Конец файла» (код 26).

Основные достоинства текстовых файлов – простота и универсальность. Фактически – это мировой стандарт представления текстовой информации. Они в основном используются там, где не имеет особого значения качество отображения информации, для обмена данными через мировые компьютерные сети, для подготовки черновиков различных изданий. Кроме того, текстовый

формат имеют файлы настроек большинство программных систем, включая MS-DOS и Windows.

2. Классификация программных средств обработки текстовой информации и текстовые документы

Для создания, просмотра и модификации (редактирования) текстовых файлов используются специальные программные средства с общим названием *текстовые редакторы*. В мире разработано огромное множество различных текстовых редакторов. Большинство из них может работать с обыкновенными текстовыми файлами. Однако, наиболее совершенные из них (включая и Microsoft Word) используют другие форматы файлов, которые, в общем случае, нельзя считать текстовыми. Такие файлы, помимо алфавитно-цифровых символов, могут содержать информацию о форматировании текста, различные графические, и другие объекты. Называются они *текстовыми документами*. Существуют, впрочем, различные программные средства преобразования (конвертирования) текстовой информации из одного формата в другой.

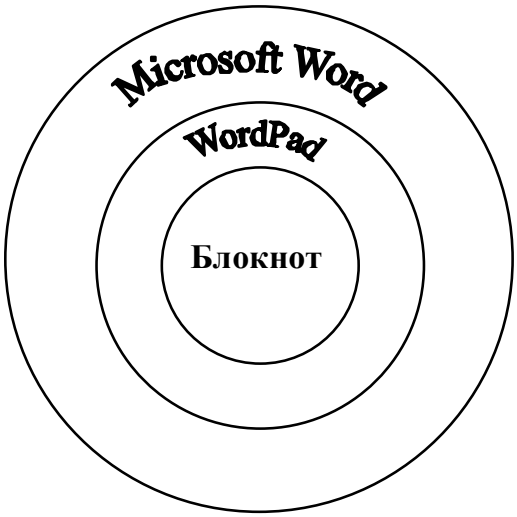
Наиболее распространенные программные средства для обработки текстовой информации условно можно разделить «по специализации» на три группы:

1. **Текстовые редакторы** (MS-DOS Editor, Блокнот, WordPad и др.) служат для создания и редактирования текстовых файлов, требования по форматированию для которых минимальны.
2. **Текстовые процессоры** (Microsoft Word, WordPerfect и др.) предназначены для обработки текстовых документов, которые вместе с текстовой информацией могут содержать и объекты другой природы, такие как рисунки, диаграммы, таблицы, формулы и т.д.
3. **Издательские системы** (Aldus PageMaker, QuarkXPress и др.). Они используются для изготовления, т.е. верстки и макетирования, полиграфической продукции особо сложной структуры (атласы, альбомы, журналы и т.д.). Текстовый и иллюстративный материал для них, как правило, подготавливается с помощью других, специализированных,

программ – текстовых и графических редакторов. Результатом работы издательской системы является *оригинал-макет*, который затем используется в типографии для тиражирования издания.

Впрочем, для подготовки к печати большинства изданий, таких как книги, буклеты и т.д. в подавляющем большинстве случае не прибегают к услугам сложных издательских систем, а используют текстовый процессор Microsoft Word.

Соотношение функциональных возможностей различных текстовых редакторов фирмы Microsoft демонстрирует следующий рисунок:



Текстовый процессор Microsoft Word помимо подготовки документов предназначенных для переноса на бумажный носитель информации, позволяет создавать так называемые *электронные документы*. Электронные (экранные) форматы документов, такие как HTML, PDF и др. не предназначены для непосредственного вывода на печатающее устройство, а используются, в основном, для публикации информации в Интернете.

Microsoft Word – это приложение Windows, предназначенное для создания, просмотра, редактирования и вывода как печатных (текстовых), так и электронных (экранных) документов. Он является одной из самых совершенных программ в классе текстовых процессоров, которая позволяет выполнять множество самых разнообразных операций над текстом и графикой.

С его помощью можно быстро и качественно подготовить любой текстовый документ – от простой записки до оригинал-макета сложного издания и Web-страницы в Интернет. Microsoft Word – один из основных элементов офисной технологии обработки текстовой информации, ставший стандартом де-факто в различных коммерческих и некоммерческих организациях.

При использовании Microsoft Word следует четко определять целевой объект – документ электронный или печатный. Для различных типов документов используют различные средства, приемы и методы. Применение неадекватных средств значительно усложняет последующие этапы работы с документом.

Как и любая другая программа, Microsoft Word обладает определенными недостатками. Однако *общепризнанных* недостатков Word не имеет, и как Windows, вообще, он является слишком *универсальной* программой. Это значит, что для его освоения необходимо приложить определенные усилия, а некоторые процедуры проще и быстрее выполнить в менее мощных, но более специализированных программах.

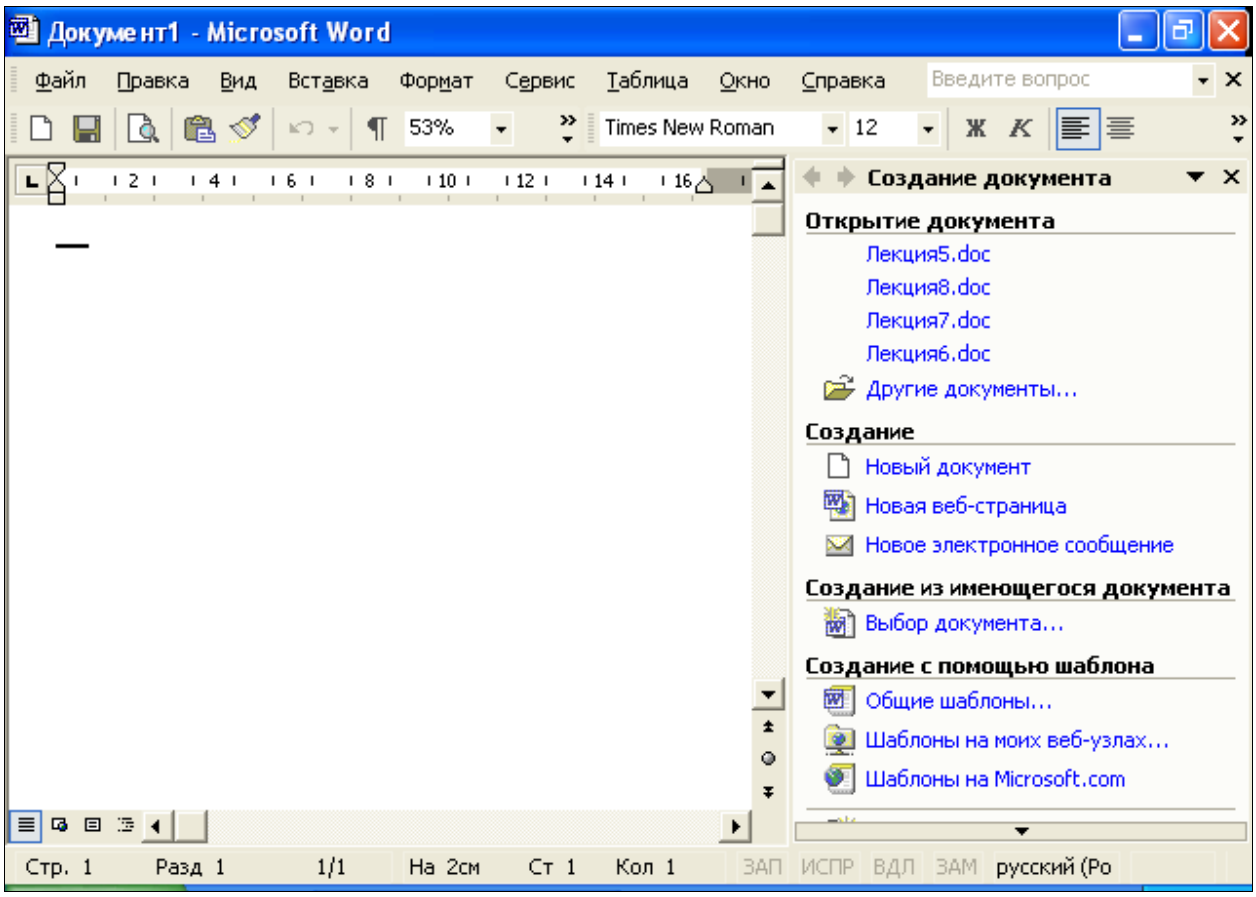
3. Запуск Microsoft Word

Текстовый процессор Microsoft Word может быть запущен на выполнение одним из стандартных способов запуска приложений Windows.

1. Через кнопку **Пуск** и основное меню системы Windows.
2. С помощью ярлыка на Рабочем столе или кнопки на панели инструментов **Быстрый запуск** (если они присутствует).
3. По двойному щелчку мышей на документе Microsoft Word – файле с расширением **.DOC**.

4. Окно Microsoft Word

После запуска Microsoft Word любым известным способом на Рабочем столе появляется основное окно приложения стандартного вида с интерфейсом типа MDI:



Помимо традиционных элементов управления оно имеет две панели инструментов – **Стандартная** и **Форматирование**, а так же разметочную линейку, расположенную ниже панелей инструментов.

5. Ввод текста

Ввод текста в документ может быть осуществлен двумя различными способами:

1. посредством набора на клавиатуре,
2. вставки из существующего файла.

Всякий текстовый документ, естественно, первоначально должен быть создан методом ввода с клавиатуры. Вводимая информация при любом способе ввода будет помещена в позицию, на которую указывает курсор – вертикальная мигающая черточка (|).

При вводе текста с клавиатуры Word автоматически осуществляет перевод курсора из конца текущей строки в начало следующей. При этом если слова не помещаются на одной строке – они переносит на новую строку.

Разбиение вводимого текста на строки в пределах абзаца осуществляется за счет автоматической вставки в конце каждой из них невидимого символа «Мягкий перевод строки». При редактировании текста или изменении длины строк мягкие переводы строк перемещаются в соответствии с характером текста, что предотвращает неверное разбиение на строки.

Тем не менее, иногда возникает необходимость в принудительном разрыве строки без начала нового абзаца. Для этого необходимо нажать сочетание клавиш **Shift** + **Enter**. При этом Word вставляет в текст документа непечатаемый символ «Жесткий перевод строки» и завершает текущую строку в не зависимости от того, достигнуто правое поле документа, или нет. Жесткий перевод строки не создает нового абзаца.

Новый абзац создается по нажатию клавиши ***Enter***. При этом если курсор находится в начале строки, и нажата клавиша **Enter** – произойдет вставка нового абзаца перед текущей строкой, а если в конце – после. Нажатие клавиши **Enter**, когда курсор находится в середине строки, вызывает ее «разлом» на две строки в позиции курсора. Ликвидация «разлома», или «склейка» строк, осуществляется посредством удаления неотображаемого символа абзаца в конце первой из них. То есть для этого необходимо переместить курсор в конец первой строки по нажатию клавиши **End** и нажать клавишу **Delete** или по нажатию клавиши **Home** поместить курсор в начало второй из них и нажать клавишу **BackSpace**. С помощью этого приема можно удалить не только неотображаемые символы, а вообще любые. Удаление одиночного символа слева от курсора осуществляется с помощью клавиши **BackSpace**, а справа от курсора – с помощью **Delete**. Если задержать в нажатом состоянии любую из клавиш **BackSpace** или **Delete**, то будет удален не один, а несколько подряд идущих символов.

Для того чтобы увидеть или, наоборот, скрыть неотображаемые символы необходимо на панели инструментов **Стандартная** нажать кнопку ¶ **Непечатаемые знаки**. При этом символ жесткого перевода строки будет отображаться как «↵», абзаца – «¶», а «самый главный» символ – пробел – «·».

Текстовый процессор Microsoft Word может находиться в одном из двух режимов ввода информации:

1. в режиме вставки (**Insert**), когда вводимый текст раздвигает вправо существующий текст, если, конечно, такой имеется, и
2. в режиме замены (**Replace**), когда вводимый текст «затирает» существующий.

В исходном состоянии, т.е. по умолчанию, включен режим вставки. Когда Word находится в режиме замены, в статусной строке индицируется «**ЗАМ**». Переключение из режима вставки в режим замены и обратно можно выполнить, как обычно, несколькими способами:

1. с помощью клавиши **Insert**,
2. двойным щелчком на индикаторе «**ЗАМ**» в строке состояния,
3. выбрать команду **Параметры** из меню **Сервис**, открыть вкладку **Правка** и установить (или сбросить) флажок **режим замены**.

Ввод прописных букв осуществляется после нажатия клавиши **Caps Lock**. При этом в правой верхней части клавиатура зажигается одноименный индикатор. Единичную прописную букву можно ввести при нажатой клавише **Shift**. Удержание клавиши **Shift** в нажатом состоянии при включенном режиме ввода прописных букв приводит к вводу строчных букв.

Переключение языка ввод на клавиатуре с русского на английский, и обратно, может быть выполнено несколькими способами:

1. на подавляющем большинстве компьютеров одновременным нажатием клавиш **Ctrl** и **Shift** (**Ctrl** + **Shift**);
2. одиночным щелчком левой кнопки мыши по индикатору языка ввода в правой части панели задач, и выбор необходимого языка в раскрывшемся меню;
3. одиночным щелчком в пределах русского или английского слова, и повторным щелчком в необходимой позиции ввода.

Последний способ переключения языка ввода доступен только при соответствующей настройке Microsoft Word.

При вводе текста иногда отдельные его части могут быть подчеркнуты красными и зелеными волнистыми линиями. Это свидетельствует об отклонении от некоторых орфографических, грамматических или стилистических правил. Красная волнистая линия свидетельствует о возможной орфографической ошибке, а зеленая – о грамматической или стилистической. Более подробную информацию о предполагаемой ошибке можно получить из диалогового окна, которое можно раскрыть по нажатию правой клавиши мыши на подчеркнутом фрагменте. *При этом следует иметь ввиду, что встроенная в Word проверка правописания лишь облегчает выявление орфографических и грамматических ошибок, однако она никогда не сможет полностью заменить тщательной редакторской и корректорской правки текста*.

6. Перемещение по документу

Перемещение позиции ввода и редактирования (или, то же самое – курсора) по документу может быть осуществлено двумя различными способами:

1. с помощью клавиатуры,
2. с помощью мыши.

С помощью клавиатуры перемещение по тексту на одну позицию вправо, влево, вверх и вниз производится посредством клавиш-стрелок управления курсором. Клавиша **Home** перемещает курсор в начало строки, а **End** – в конец. Клавиши **Page Up** и **Page Down** перемещают окно редактирования на один экран, соответственно, к началу или концу текстового документа. Сочетание клавиш **Ctrl** + **Home** перемещает курсор в начало документа, а **Ctrl** + **End** – в конец. **Ctrl** + **Page Up** и **Ctrl** + **Page Down** перемещают окно редактирования на предыдущую, или следующую *страницу* текстового документа, соответственно. Одновременное нажатие клавиш **Shift** + **F5** перемещает курсор к месту внесения в документ последних изменений.

Перемещение курсора по тексту с помощью мыши выполняется более естественным образом. Если необходимый фрагмент текста виден в рабочей области окна документа Word, то необходимо просто поместить указатель

мыши в соответствующую позицию и щелкнуть левой кнопкой мыши. Если же необходимый фрагмент текста в окне отсутствует, то с помощью полос прокрутки его необходимо поместить туда, и далее, просто щелкнуть левой кнопкой мыши в нужной позиции. *При прокрутке текста с помощью полос прокрутки положение курсора не изменяется*. В связи с этим довольно распространенной ошибкой является то, что после отображения в окне нужного фрагмента сразу начинается ввод. При этом ввод информации будет происходить в той позиции, где остался курсор, а не там, где предполагалось.

Переход к определенному участку документа можно выполнить так же с помощью *кнопок перехода*, расположенных ниже вертикальной полосы прокрутки.

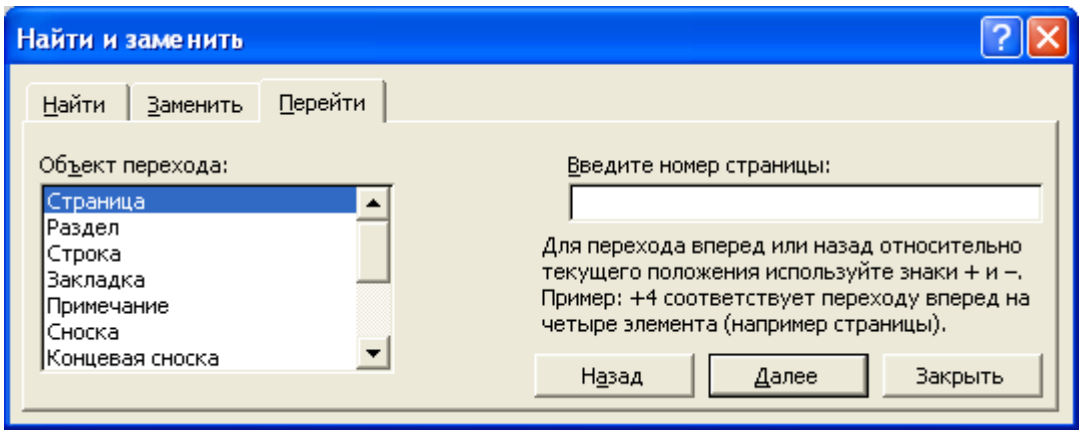
В отличие от полос прокрутки при использовании кнопок перехода производится перемещение курсора. По умолчанию при помощи кнопок перехода осуществляется переход к предыдущей или следующей странице документа. Однако эту установку можно изменить с помощью кнопки **Выбор объекта перехода**. При щелчке на ней левой кнопкой мыши на экране отображается палитра объектов перехода, позволяющая указать, к какому объекту будет осуществляться переход при нажатии кнопки перехода.

Наведение указателя мыши на любую из кнопок палитры объектов перехода вызывает появление описания соответствующего объекта в нижней части палитры. Нажатие кнопки приводит к выбору соответствующего объекта в

качестве объекта перехода. При выборе любого объекта перехода, кроме страницы, кнопки перехода окрашиваются в синий цвет, что указывает на выбор нестандартного объекта перехода. Кнопки перехода могут оказаться полезными, когда необходимо просмотреть содержащиеся в документе объекты, представляющие информацию определенного рода, например, рисунки, таблицы, рецензии, сноски и т.д. *Если требуется узнать, какой объект выбран в качестве текущего объекта перехода, достаточно навести указатель мыши на одну из кнопок перехода и дождаться появления всплывающей подсказки*.

Если необходимо перейти к определенному объекту в документе, это можно сделать значительно быстрее с использованием команды **Перейти**. Она позволяет выполнить поиск определенных элементов документа, или прейти к определенной таблице, рисунку, рецензии или другому объекту. Переход или поиск выполняется с использованием окна диалога **Найти и заменить**, которое имеет три вкладки: **Найти**, **Заменить** и **Перейти**. Открыть окно **Найти и заменить** на вкладке **Перейти** можно одним из следующих способов:

1. выбрать команду **Перейти** в меню **Правка**,
2. нажать сочетание клавиш **Ctrl** + **G**,
3. нажать кнопку → **Перейти** на палитре объектов перехода.



В окне диалога **Найти и заменить** необходимо выбрать тип объекта перехода и указать его номер или имя. С помощью этого окна можно также перейти на предыдущий или следующий объект выбранного типа. При использовании команды **Перейти** Word устанавливает в качестве объекта перехода по нажатию кнопок перехода, расположенных ниже вертикальной полосы прокрутки, тип объекта, по которому осуществлялся последний переход.

Перемещение по документу можно осуществлять также с использованием функций поиска и замены. *Каким бы способом не выполнялось перемещение курсора, оно всегда выполняется только в пределах текущего документа*.

7. Выделение текста

После завершения ввода текста документа может понадобиться внести в него некоторые поправки. Но, прежде чем вносить изменения в определенный текстовый фрагмент, его необходимо предварительно выделить (пометить, маркировать). Выделенная область указывает Word, с какой частью текста необходимо выполнить те или иные операции. Существует несколько способов выделения текста:

1. с помощью только клавиатуры,
2. с помощью только мыши,
3. с помощью клавиши **Shift** и щелчка мыши – комбинированный.

Выделение фрагмента текста с помощью клавиатуры выполняется следующим образом:

1. курсор перемещается в начало выделяемого фрагмента,
2. при нажатой клавише **Shift** любым известным способом курсор перемещается в конец выделяемого фрагмента.

Выделение фрагмента текста с помощью мыши также можно выполнить двумя различными методами:

1. методом протяжки,
2. методом щелчка.

Выделение фрагмента текста методом протяжки является наиболее интуитивно понятным способом. При этом необходимо:

1. поместить курсор в начало выделяемого фрагмента,
2. при нажатой левой кнопке мыши переместить курсор в конец выделяемого фрагмента.

Таким образом, можно выделить любую часть текста: от одного символа до всего документа. Когда при протягивании указатель мыши достигает одного из краев окна редактирования, одновременно с расширением выделяемой области выполняется автоматическая прокрутка документа в соответствующем направлении. *Для выделения прямоугольной области текста в пределах нескольких строк вместе с перемещением курсора необходимо держать нажатой клавишу* ***Alt***.

Методом щелчка можно выделить:

1. одно слово – при двойном щелчке на нем,
2. одно предложение – при щелчке с нажатой клавишей **Ctrl**,
3. один абзац – при тройном щелчке в пределах абзаца,
4. одну строку – при щелчке слева от строки,
5. весь документ – при тройном щелчке на левом поле документа, или один щелчок там же при нажатой клавише **Ctrl**.

Для выделения текста комбинированным способом (вместе с клавишей **Shift**) необходимо выполнить следующую последовательность действий:

1. поместить курсор в начало выделяемого фрагмента любым известным способом,
2. при нажатой клавише **Shift**, щелкнуть левой кнопкой мыши в конце выделяемого фрагмента.

Этот прием может оказаться особенно полезным, если необходимо «передвинуть» конец выделяемого фрагмента текста – для этого просто нужно при нажатой клавише **Shift** щелкнуть левой кнопкой мыши в новом месте.

Для снятия выделения достаточно переместить курсор из текущей позиции любым известным способом.

8. Редактирование текста

После выделения подлежащего редактированию участка текста его можно изменять. При этом основными операциями по редактированию текста являются следующие:

1. удаление,
2. вырезание,
3. вставка,
4. замена,
5. копирование,
6. перемещение.

Каждая из них может быть выполнена несколькими различными способами, и в каждой конкретной ситуации необходимо использовать наиболее простой и подходящий из них.

Наиболее простой способ **удаления** выделенного фрагмента текста состоит в нажатии клавиши **Delete** или **BackSpace**. Сочетания клавиш **Shift** + **Delete** приводит к **вырезанию** выделенного фрагмента текста. Операция вырезать удаляет текст из документа и помещает его в буфер обмена (**Clipboard**) – некоторый служебный участок оперативной памяти компьютера. Позже помещенный в буфер обмена фрагмент текста можно будет сколько угодно раз вставить в другое место документа. Если есть выделенный фрагмент, то становится доступной как кнопка  **Вырезать** панели инструментов **Стандартная**, так и одноименная команда из контекстного и основного меню **Правка**.

Если буфер обмена не пуст, то находящийся в нем объект можно **вставить** в то место документа, на которое указывает курсор. При этом становятся доступными кнопка  **Вставить** на панели инструментов **Стандартная**, а так же одноименные команды из контекстного и основного меню **Правка**. Быстро вставить объект из буфера обмена в документ можно так же при помощи сочетания клавиш **Ctrl** + **Insert**. Операция вставить не очищает буфер обмена, так что один и тот же объект можно вставлять много раз.

Операция **заменить** состоит в удалении помеченного объекта, и вставке на его место объекта из буфера обмена. Заменять выделенный объект можно не только из буфера обмена, но и при вводе с клавиатуры.

Операция **копировать** помещает выделенный фрагмент текста в буфер обмена без его удаления из документа. Чтобы скопировать выделенный объект в буфер обмена необходимо нажать сочетание клавиш **Ctrl** + **Insert**, или щелкнуть на кнопке  **Копировать** на панели инструментов **Стандартная**, или выбрать одноименную команду контекстного меню или основного меню **Правка**.

Копирование и перемещение фрагментов текста можно выполнить и без использования буфера обмена. Для этого необходимо:

1. щелкнуть левой кнопкой мыши на выделенном фрагменте,
2. при нажатой левой кнопке мыши перетащить фрагмент текста в новое место.

Если при этом будет нажата клавиша **Ctrl**, то будет выполнено не перемещение, а копирование текста. При выполнении операции копирования перетаскиваемый фрагмент будет дополнительно помечен знаком «+» в квадратике. Копирование и перемещение выделенных фрагментов текста удобно выполнять, если на экране одновременно видны и исходный и результирующий участки документа. Однако перемещение и копирование фрагментов текста можно выполнять не только в пределах одного, но и разных документов.

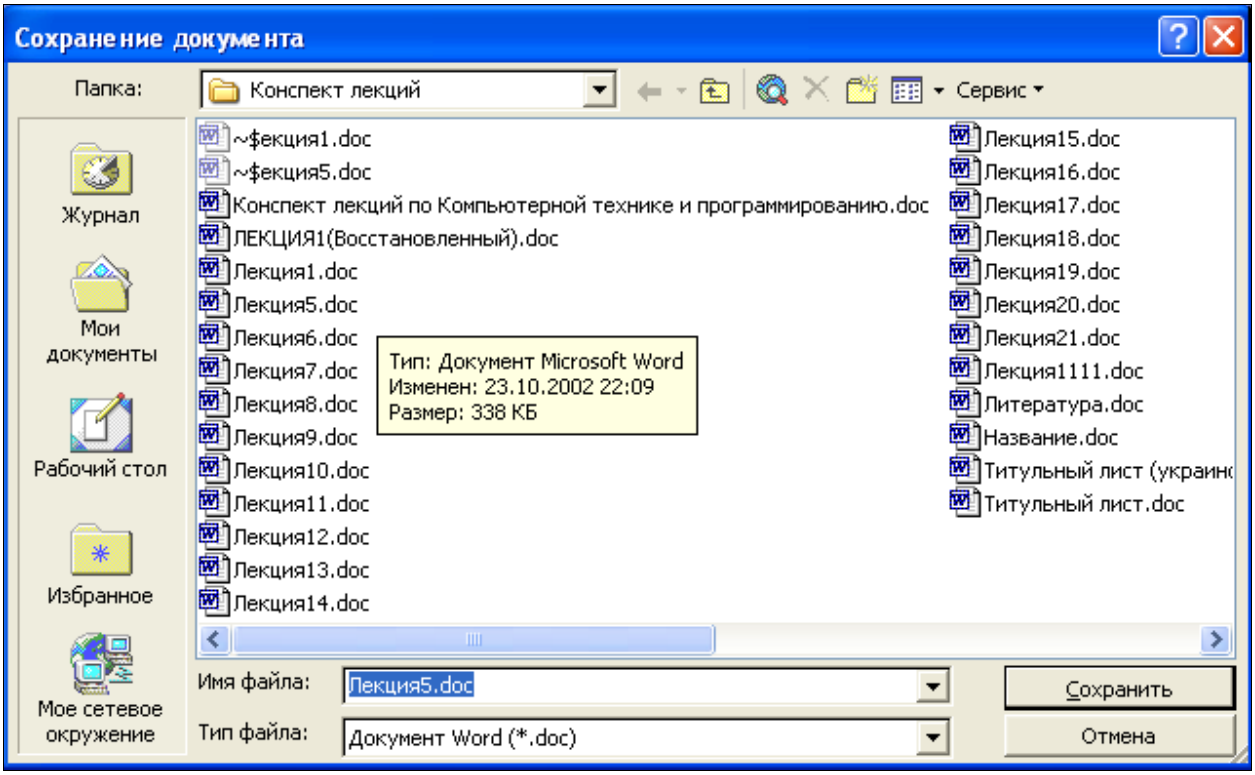
9. Отмена и возврат действий

Иногда в процессе работы над документом может случиться так, что после внесения в него некоторых изменений, возникнет необходимость отказаться от них. Для отмены последнего внесенное в документ изменение необходимо нажать кнопку  **Отменить** на панели инструментов **Стандартная** или выбрать одноименную команду в основном меню **Правка**. Сразу после того, как действие было отменено, в меню **Правка** ниже команды

Отменить появляется команда **Вернуть**. Для возврата последнего отмененного изменения (т.е. для «отмены команды **Отменить**»), следует выбрать эту команду или воспользоваться кнопкой **Вернуть** на панели инструментов **Стандартная**. Если необходимо отменить или вернуть не одно, а несколько действий, то необходимо щелкнуть на маленькой кнопке со стрелкой, расположенной справа от кнопок **Отменить** или **Вернуть**, и из раскрывшегося списке выбрать необходимое.

10. Сохранение документа

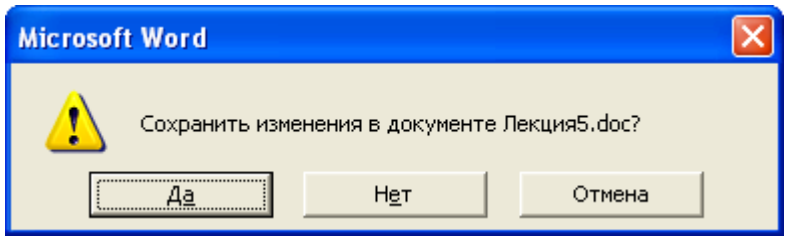
Чтобы документы, созданные средствами Microsoft Word, можно было использовать в дальнейшем или передавать другим пользователям, их необходимо сохранять в дисковых файлах. Для сохранения документа нужно нажать кнопку **Сохранить** на панели инструментов **Стандартная**, или выбрать одноименную команду из основного меню **Файл**. При этом если документ сохраняется первый раз, на экране будет отображено окно диалога **Сохранение документа**.



В этом окне необходимо присвоить имя новому документу и указать папку, в которой он будет записан. Перемещение курсора между полями в пределах окна можно выполнить с помощью клавиши **Tab** или с помощью мыши. Если документ сохраняется не первый раз, окно диалога **Сохранение документа** не открывается. Для сохранения документа под новым именем или в другой папке необходимо выбрать команду **Сохранить как…** из основного меню **Файл**. При этом также будет открыто окно диалога **Сохранение документа**.

11. Завершение работы

Для завершения работы Microsoft Word необходимо выбрать команду **Выход** из основного меню **Файл** или щелкнуть на кнопке **Закрыть** в строке заголовка основного окна приложения, если открыт только один документ. Щелчок на кнопке **Закрыть** на панели меню активного документа приводит к закрытию только этого документа. При этом если не были сохранены последние внесенные в документы изменения, Word выдаст соответствующее предупреждение для каждого из таких документов.



Для сохранения изменений при закрытии документа следует щелкнуть на кнопке **Да**. При включенном режиме отображения **Помощника** (**Office Assistant**) подобного рода сообщение будет отображаться **Помощником**.

ЛЕКЦИЯ №4

СОЗДАНИЕ, РЕДАКТИРОВАНИЕ И ФОРМАТИРОВАНИЕ ЭЛЕКТРОННЫХ ТАБЛИЦ

План

1. Введение
2. История создания электронных таблиц
3. Организация данных в Microsoft Excel
4. Адресация ячеек
5. Особенности интерфейса
6. Режимы работы
7. Завершение работы

1. Введение

Электронные таблицы (табличные процессоры) – это программы, которые предназначены для обработки информации, представленной в табличном виде, т.е. в виде строчек и столбцов, на пересечении которых находятся ячейки.

Основная идея электронных таблиц состоит в том, что одна часть ее ячеек содержит исходные данные, а другая – формулы, по которым эти данные пересчитываются. В результате этого всякое изменение содержимого какой-либо ячейки исходных данных приводит к немедленному изменению содержимого ячеек с формулами – т.е. содержимое электронной таблицы обновляется (пересчитывается) автоматически, и всегда остается актуальным.

Столь простая, и в тоже время, абсолютно гениальная идея послужила толчком к созданию огромного количества программ этого класса – в мире, наверное, нет ни одного компьютера, на котором бы не использовалась та, или иная, электронная таблица. Наиболее широкое распространение они получили в области выполнения различного рода экономических расчетов, поскольку являются естественным продолжением идеи хорошо известных всем экономистам «пустографок». При этом переход от традиционной, «бумажной

пустографки», к ее электронному эквиваленту происходит совершенно легко и естественно. А, однажды заполнив такую «электронную пустографку» исходными данными и формулами в дальнейшем ее можно только «эксплуатировать» – заносить исходные данные, а результаты будут получаться автоматически, и притом без всяких ошибок усталости и невнимательности, свойственных ручному счету. Однако область применения электронных таблиц вовсе не ограничивается только экономическими расчетами. Благодаря наличию в них большого количества самых разнообразных функций, а также средств наглядного представления информации в виде различного рода графиков и диаграмм, они нашли самое широкое применение и в других областях выполнения расчетов, таких, например, как инженерные, научно-технические и т.д. Если сравнить электронную таблицу с таким широко распространенным инструментом для выполнения вычислений как калькулятор, то можно смело сказать, что последний выполняет только *арифметические* вычисления, в то время как электронная таблица – также и *алгебраические*, в том смысле что она может оперировать не только *константами*, как калькулятор, но и *переменными* – содержанием ячеек.

2. История создания электронных таблиц

Самая первая электронная таблица – VisiCalc – была создана в 1979 году Дэниелом Бриклином и Робертом Фрэнкстоном и принесла своим создателям только в течение первого года продаж около двух миллионов долларов чистой прибыли. А началось все с того, что опыт работы в компьютерной индустрии выпускника Массачусетского технологического института (МТИ) Дэниела Бриклина навел его на мысль, что, не плохо разбираясь в компьютерах, он при этом совершенно ни чего не понимает в бизнесе. Это и побудило его пойти на курсы в Гарвардскую школу бизнеса. Во время занятий на этих курсах ему приходилось выполнять много однообразных вычислений, основная специфика которых заключалась в том, что при изменении какой-либо одной цифры необходимо было пересчитывать все, зависящие от нее, величины. Малейшая ошибка при этом могла испортить всю работу, которая записывалась на

большие листы тщательно разлинованной бумаги – *Spreadsheets* – развернутые листы, или по-русски – «простыни» «пустографки». При этом программистский опыт Бриклина натолкнул его на мысль, что все эти однообразные и нудные «жонглирования» цифрами неплохо было бы «поручить» компьютеру. Своими соображениями он поделился с Робертом Фрэнкстоном, которого эта идея тоже вдохновила, и они вместе приступили к ее реализации. В результате к весне 1979 года идея была воплощена в законченную программу, которую ее создатели назвали VisiCalc (Visible Calculator – визуальный калькулятор). Изначально она была написана для компьютера Apple-2 и в течение первого года после начала продаж (с августа 1979 года) разошлась тиражом около 100 тыс. экземпляров по цене 200 долл. за штуку.

Первая же электронная таблица для компьютеров типа IBM PC была создана Митчеллом Кэпором спустя три года – в конце 1982 году – и называлась Lotus 1-2-3. При этом на ее рекламу было истрачено 1 млн. долл. Но эти расходы окупились очень скоро – уже через полгода было продано 60 тыс. копий по цене 495 долл. за штуку. Одним из наиболее распространенных табличных процессоров в предшествующие десятилетия на территории бывшего Советского Союза был SuperCalc фирмы Computer Associates – «облегченный», по сравнению с Lotus 1-2-3, вариант электронных таблиц для операционной системы MS-DOS.

В настоящее время, ведущее положение на мировом рынке программных продуктов этого класса занимает электронная таблица Excel корпорации Microsoft, первая версия которой была выпущена в 1985 году. В 1993 году появилась 5-я версия Excel, ставшая первым приложением знаменитого пакета Microsoft Office. В дополнение к традиционным табличным вычислениям, в основном, в области экономики и финансов, Microsoft Excel позволяет строить различные диаграммы, графики и рисунки, проводить сложную статистическую и математическую обработку данных, обмениваться информацией с наиболее распространенными базами данных, создавать Web-страницы и т.д. Средствами табличного процессора Microsoft Excel можно производить полный цикл

обработки информации – от сбора, регистрации и, собственно, обработки до отображения ее в наиболее презентабельном виде, достигая при этом поистине ошеломляющих результатов.

3. Организация данных в Microsoft Excel

Любая программа, и Microsoft Excel в том числе, может обрабатывать данные только вполне определенного вида (формата, структуры). Основными объектами данных, с которыми может работать Microsoft Excel, являются следующие:

1. *рабочая книга*,
2. *лист*,
3. *строка*,
4. *столбец*,
5. *ячейка*,
6. *диапазон* и
7. *список*.

Документ (т.е. объект обработки) приложения Microsoft Excel называется *рабочая книга*. Каждая рабочая книга имеет собственное имя, построенное в соответствии со стандартами операционной системы Windows, и хранится в отдельном дисковом файле. По умолчанию, новым рабочим книгам Microsoft Excel присваивает имена **Книга1**, **Книга2** и т.д. При их записи на диск к этому имени по умолчанию добавляется расширение **.XLS**, так что на диске будут записаны, соответственно, файлы **Книга1.XLS**, **Книга2.XLS** и т.д. В Microsoft Excel допускается одновременная работа с несколькими рабочими книгами, при чем каждая из них открывается в отдельном окне документа. Количество одновременно открытых рабочих книг ограничивается только наличием свободных ресурсов компьютера. При этом *активной*, или *текущей*, может быть только одна из них.

Каждая рабочая книга состоит из *листов* двух типов:

1. *Рабочий лист* (или просто лист) образует основное рабочее пространство пользователя, которое может состоять из таких компонентов:

а) *ячеек*, организованных в виде *строк* и *столбцов*,

б) одной или нескольких диаграмм,

в) стандартных элементов управления, таких как кнопки, флажки, переключатели и т.д., и

г) прочих связанных и внедренных объектов из других приложений.

Все компоненты, кроме а), необязательны и могут отсутствовать на конкретном рабочем листе – они туда помещаются явным образом. По умолчанию рабочим листам присваиваются имена **Лист1**, **Лист2** и т.д.

2. *Лист диаграмм* предназначен для размещения на нем диаграмм. На одном листе диаграмм может размещаться одна, или несколько, диаграмм. Диаграммы можно свободно перемещать между листами диаграмм и рабочими листами в любом направлении. Стандартные имена листов диаграмм – **Диаграмма1**, **Диаграмма2** и т.д.

Количество листов в рабочей книге может быть произвольным – оно ограничено только наличием свободных ресурсов на компьютере. По умолчанию каждая, вновь создаваемая рабочая книга, содержит три рабочих листа – **Лист1**, **Лист2** и **Лист3**.

Каждый рабочий лист состоит из 65 536 = 2^{16} пронумерованных *строк*, начиная с 1, и 256 = 2^8 *столбцов*, проиндексированных буквами *латинского алфавита*, начиная с A, и заканчивая IV. На пересечении строк и столбцов находятся *ячейки*, общее количество которых – 16 777 216 = 65 536 × 256.

Ячейка – это элементарная (неделимая) и однозначно адресуемая единица информации, обрабатываемой с помощью электронной таблицы.

Каждая ячейка рабочего листа может быть:

1. пустая, или
2. непустая.

Непустая ячейка, в свою очередь, может содержать:

а) *формулу*, или

б) *константу*.

Константы бывают:

1. *Числовые*, которые могут состоять только из цифр и, возможно, некоторых специальных знаков, таких как + - ( ) , / $ % . E e.
2. *Текстовые* – это константы, которые Excel не смог распознать как константы другого типа. Они могут содержать любые символы из допустимого набора, длиной не более 32 767 знаков. Текстовыми константами могут быть и числа, если при вводе им предшествует апостроф, например, '1234. Это могут быть, например, табельные номера сотрудников. Естественно, что такие константы не могут участвовать в арифметических вычислениях.
3. *Логические* – могут принимать только одно из двух взаимоисключающих значений: ИСТИНА или ЛОЖЬ. Они используются как индикатор наличия или отсутствия какого-либо признака или события, а также могут являться параметрами или возвращаемыми значениями некоторых функций. Во многих случаях вместо этих значений используются цифры 1 и 0, соответственно.
4. *Ошибки* – служат для индикации некоторых нештатных ситуации, возникших во время работы Excel, например деление на нуль. Значения этого типа констант начинаются с символа «#». Более детальную информацию о причине возникновения конкретной ошибки можно получить в справочной системе Microsoft Excel, осуществив поиск по типу отображаемой ошибки.

Кроме перечисленных выше основных типов данных Microsoft Excel поддерживает еще два производных:

5. *Дата и время суток* (или *Календарный*) – это тот же числовой тип данных, в котором целая часть используется для хранения количества дней прошедших с 1 января 1900 года, а дробная – это часть суток, прошедшая с полуночи.
6. *Массив*, собственно, не являются конкретным типом данных, а только образует организованное множество ячеек или констант любого типа. В Microsoft Excel массив рассматривается как единый элемент, к которому в

целом могут быть применены математические, логические и другие типы операций.

Формулы в Microsoft Excel служат для получения новых значений, т.е. являются основным инструментом преобразования информации. Всякая формула состоит из 2-х частей:

1. знака равно (=),
2. за которым следует *выражение*.

То есть, синтаксис формулы можно представить следующим образом:

```
= <выражение>
```

и означает, что результат вычисления выражения будет присвоен в качестве *значения* ячейке, в которую эта формула записана. Выражение, в свою очередь, состоит из:

1. *констант*,
2. *адресов (или ссылок на) ячейки*,
3. *функций*,
4. соединенных *знаками операций*.

При этом *круглые скобки* в выражении позволяют изменить стандартный (естественный) порядок выполнения операций. Например, формула `= 1` присваивает в качестве значения ячейки, в которой она записана, константу числового типа 1, а `= A1 + 1` – записывает в текущую ячейку значение ячейки `A1`, увеличенное на `1`.

Кроме констант или формул с любой ячейкой может быть также связано *примечание* – дополнительный текст поясняющий, например, ее содержание. Индикатор наличия примечания – маленький красный треугольник в верхнем правом углу ячейки. При перемещении указателя мыши в пределы ячейки, содержимое примечания отображается в отдельном окне примечания.

4. Адресация ячеек

Для использования значений, над которыми требуется выполнить некоторые вычисления, необходимо указать расположение ячеек, которые содержат эти значения. Расположение ячеек в Microsoft Excel однозначно определяется их *адресами* (*ссылками*, *координатами* и т.д.). Для адресации ячеек в пределах одного рабочего листа используется два *стиля адресации*, которые, соответственно, называются:

1. **A1** – указывается имя столбца и номер строки,
2. **R1C1** – указывается символ R (от английского Row – строка), номер строки, символ C (от английского Column – столбец) и номер столбца.

Например, ячейка на пересечении 3-й строки и 4-го столбца в стиле **A1** имеет адрес D3, а в стиле **R1C1** – R3C4. *По умолчанию в Microsoft Excel используется стиль адресации ячеек* ***A1***. Стиль же ссылок **R1C1** довольно часто используется в *макросах*. Для того чтобы включить или выключить стиль ссылок **R1C1**, необходимо установить или сбросить, соответственно, флажок **Стиль ссылок R1C1** на вкладке **Общие** диалогового окна **Параметры**, которое раскрывается по одноименной команде из меню **Сервис**.

При изменении стиля ссылок Microsoft Excel автоматически производит соответствующие изменения в ссылках на ячейки во всех формулах рабочей книги. Визуальным признаком того, что Excel использует стиль адресации ячеек ***R1C1****, является то, что в качестве заголовков столбцов используются цифры, а не буквы латинского алфавита, как в стиле* ***A1***.

Кроме этого, в Excel можно задавать ссылки на ячейки других листов той же книги, другой книги, а также на данные из других приложений.

Ссылки на ячейки других рабочих книг называются ***внешними****, ссылки на данные из других приложений –* ***удаленными***.

Ссылки на ячейки в пределах рабочей книги включают имя листа и адрес ячейки на листе. Имя рабочего листа отделяется от адреса ячейки символом восклицательного знака (!).

 *Если имя листа содержит пробелы или начинается с цифры, оно заключается в одиночные апострофы, в противном случае апострофы не используются.*

Например:

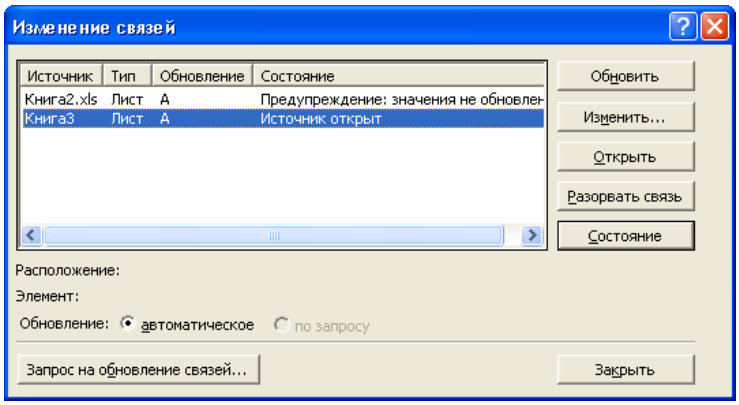
Накладная!D3 – адрес ячейки **D3** на рабочем листе **Накладная**.

При необходимости анализа данных из одних и тех же ячеек на нескольких листах одной и той же рабочей книги в Microsoft Excel используются так называемые *трехмерные ссылки*. Трехмерная ссылка включает в себя ссылку на ячейку, или диапазон ячеек, перед которой указывается диапазон имена листов. *В трехмерной ссылке Excel использует все рабочие листы, указанные между первым и последним*. Например, формула **=СУММ(Лист2:Лист5!A1)** суммирует все значения, содержащиеся в ячейке A1 на всех рабочих листах от Лист2 до Лист5 включительно.

В Microsoft Excel можно создавать ссылки не только между различными листами одной и той же книги, но и организовать иерархию связанных книг.

 *Если ячейки, на которые имеются ссылки, изменяются, Excel автоматически обновляет ссылки только для открытых книг, содержащих связи. Если же зависимая книга закрыта, связи можно обновить вручную, выполнив следующие действия*:

1. Выполнить команду **Правка ⇨ Связи**. Недоступность пункта **Связи** означает, что данная рабочая книга не содержит связанных данных.
2. В раскрывшемся при этом окне диалога **Изменение связей**:



а) В списке выбрать источник для связанного объекта. Для выделения нескольких связей необходимо выбрать их при нажатой клавише **CTRL**.

б) Щелкнуть по кнопке **Обновить**.

Microsoft Excel отображает ссылками в формулах на другие книги двумя способами, в зависимости от состояния исходной книги (той, что предоставляет данные для формулы):

а) *открыта* она или

б) *закрыта*.

Внешний адрес ячейки в *открытой* рабочей книге содержит:

1. имя рабочей книги,
2. имя рабочего листа и
3. адрес ячейки.

Имя рабочей книги берется в квадратные скобки, имя листа следует за именем книги без разделителей, которое, в свою очередь, отделяется от адреса ячейки восклицательным знаком, например:

[Товары.XLS]Накладная!D3 – адрес ячейки **D3** на рабочем листе **Накладная** открытой рабочей книги **Товары.XLS**.

Когда книга *закрыта*, ссылка должна включать дополнительно полный путь. Например:

’C:\Мои документы\[Товары.XLS]Накладная’!D3 – адрес ячейки **D3** на рабочем листе **Накладная** рабочей книги **Товары.XLS**, которая находится в папке **Мои Документы** на диске **C**.

Если имя рабочего листа, книги или путь содержит символы, не являющиеся буквами, необходимо заключить их в одиночные кавычки.

Относительная и абсолютная адресация

Оба способа адресации указывают на одни и те же ячейки. Их различие проявляется только в двух случаях:

1. при копировании ячеек, содержащих формулы, и
2. изменении структуры рабочего листа.

Относительная адресация основана на том, что ссылки на ячейки с исходными данными создаются относительно ячейки, содержащей формулу. Это означает, что при добавлении или удалении строк, столбцов или отдельных ячеек, а также при копировании формулы в другие ячейки ссылки в каждой копии изменяются таким образом, чтобы сохранились те же соотношения адресов, что и в исходной формуле. То есть, в формуле учитывается «расстояние» между ячейкой, содержащей формулу, и ячейкой, на которую в этой формуле есть ссылка.

При *абсолютной* адресации ссылка на ячейку содержит букву столбца и номер строки, перед которыми стоит знак доллара ($). При этом предполагается, что в результате копирования или изменении структуры рабочего листа ссылка не изменяется, и будет указывать на ту же, что и ранее, ячейку.

Смешанная адресация, т.е. комбинация относительной и абсолютной предполагает «фиксацию» только одной из двух «координат» ячейки. Циклическое изменение способа адресации во время ввода или редактирования с относительного на абсолютный и, далее, на смешанный выполняется путем последовательных нажатий на клавишу **F4**.

Результаты выполнения копирования ячейки, в которую записана простейшая формула (например, =E7), при различных способах адресации представлены на следующих рисунках.

=C5	=D5	=E5	=F5	=G5
=C6				=G6
=C7		=E7		=G7
=C8				=G8
=C9	=D9	=E9	=F9	=G9

=$E$7	=$E$7	=$E$7	=$E$7	=$E$7
=$E$7				=$E$7
=$E$7		=$E$7		=$E$7
=$E$7				=$E$7
=$E$7	=$E$7	=$E$7	=$E$7	=$E$7

=C$7	=D$7	=E$7	=F$7	=G$7
=C$7				=G$7
=C$7		=E$7		=G$7
=C$7				=G$7
=C$7	=D$7	=E$7	=F$7	=G$7

=$E5	=$E5	=$E5	=$E5	=$E5
=$E6				=$E6
=$E7		=$E7		=$E7
=$E8				=$E8
=$E9	=$E9	=$E9	=$E9	=$E9

⚠ *По умолчанию в стиле* ***A1*** *используется относительная адресация, в то время как в* ***R1C1*** *– абсолютная.*

При работе с электронными таблицами довольно часто возникает ситуация, когда с некоторой совокупностью ячеек необходимо работать как с единым целым, например при их удалении, копировании, перемещении и т.д. В этом случае применяются так называемые *диапазоны ячеек*. Для указания диапазона ячеек используется символ двоеточия (:), помещенный между ссылками на первую и последнюю ячейки диапазона, например, `B2:D4`. *Диапазон может включать только смежные ячейки таблицы*. Чтобы ссылка указывала на целые столбцы или целые строки, следует использовать диапазон, содержащий в обеих частях имена столбцов или номера строк, например, `B:C, 3:5`. Такое обозначение диапазона позволяет автоматически включать в обработку все ячейки данных строк или столбцов, даже если впоследствии возникнет необходимость в добавлении или удалении последних. *Вместо символа двоеточия при вводе можно использовать и символ точки (.).*

Для разделения элементов *списка*, состоящего из отдельных ячеек, и (или) диапазонов ячеек, используется символ «Разделитель элементов списка», например, `2:3,B5`.

⚠ *В разных версиях и вариантах локализации операционной системы Microsoft Windows символ «Разделитель элементов списка» может иметь различное значение.*

Так, например, в русифицированной версии операционной системе Windows XP по умолчанию – это символ точки с запятой (;), а в панъевропейской – запятая (,). Его установка, например, в операционной системе Windows XP осуществляется по такому алгоритму:

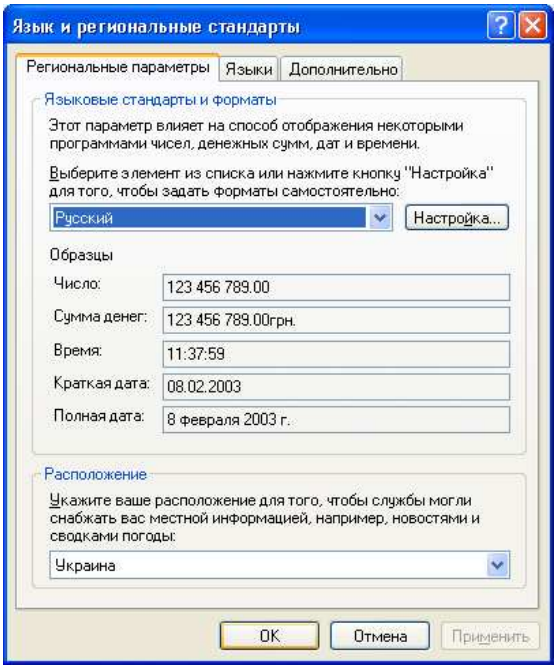
1. Выполнить команду **Пуск ⇨ Панель управления**.
2. В раскрывшемся в результате окне **Панель управления**, при отображении его:
   а) в классическом виде выбрать значение **Языки и региональные стандарты**, а

б) с разбивкой по категориям возможностей – **Дата, время языки и региональные стандарты**.

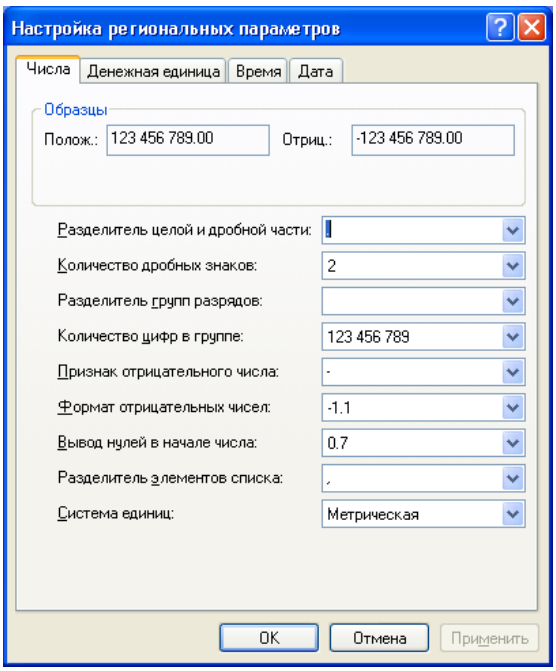
б.1) В раскрывшемся окне **Дата, время языки и региональные стандарты** выбрать:

- ❒ **Изменение формата отображения чисел, даты и времени**, или
- ❒ **Языки и региональные стандарты**.

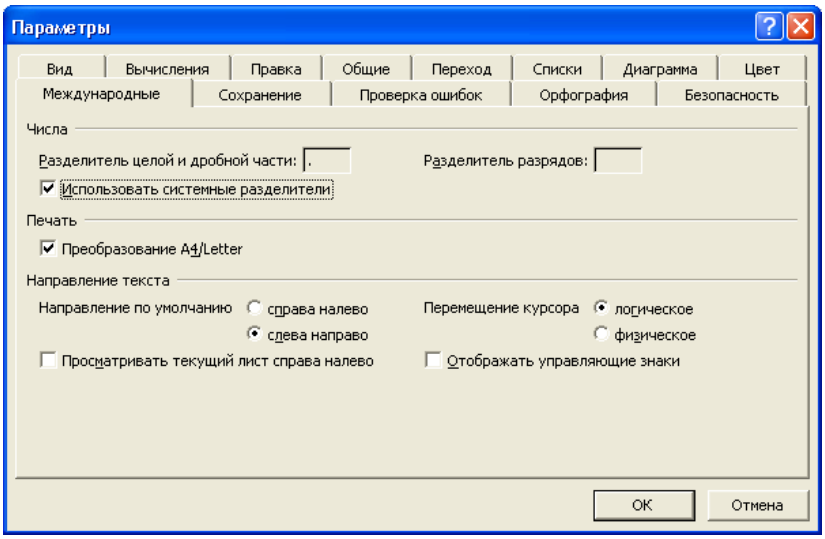
3. В окне диалога **Языки и региональные стандарты** нажать кнопку **Настройка**.



4. В поле со списком **Разделитель элементов списка** окна диалога **Настройка региональных параметров** указать необходимый символ-разделитель элементов списка.

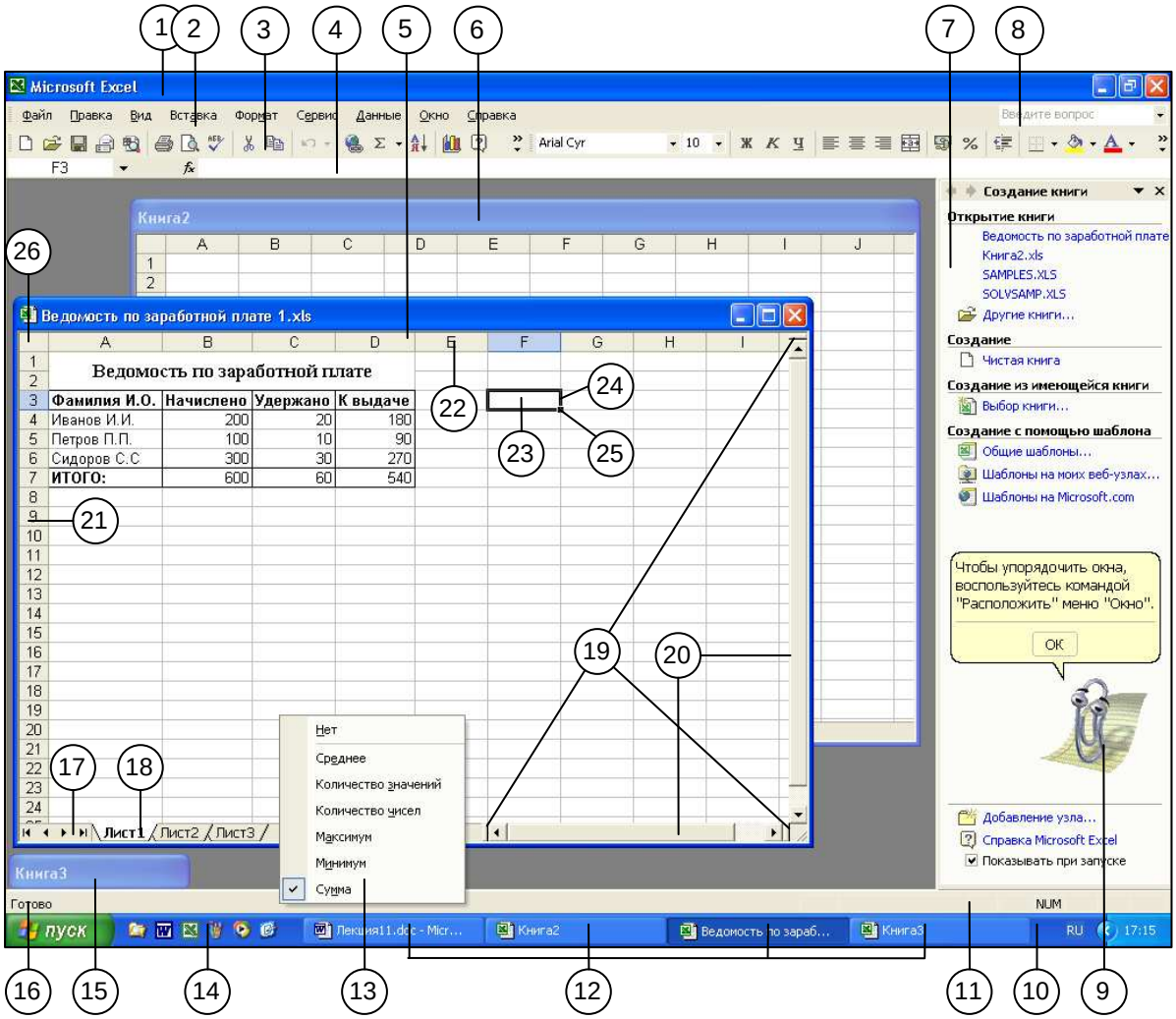


В этом окне диалоговом можно установить так же символ-разделитель целой и дробной части числа, количество знаков в дробной части числа, количество цифр в группе, и другие параметры отображения числовых величин, даты, времени, а так же денежных единиц для операционной системе Windows в целом. Отказаться от использования системных (глобальных) разделителей, и определить свои, можно на вкладке **Международные** окна диалога **Параметры**, которое открывается по команде **Сервис ⇨ Параметры ⇨ Международные**:



5. Особенности интерфейса

После запуска Microsoft Excel на Рабочем столе Windows появляется основное окно этого приложения. По умолчанию в нем открывается единственное окно документа, в котором отображается вновь созданная рабочая **Книга1** с тремя рабочими листами – **Лист1**, **Лист2** и **Лист2**. На рис. 1.1 представлено основное окно приложения Microsoft Excel, в котором одновременно ведется обработка трех рабочих книг.



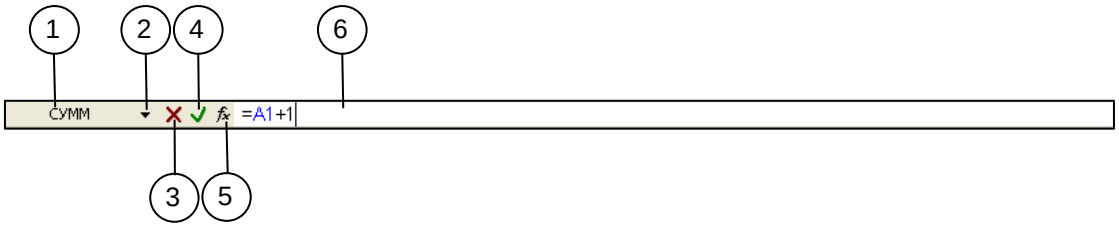
1. Заголовок окна приложения
2. Основное меню приложения
3. Панели инструментов **Стандартная** и **Форматирование**
4. Строка (панель) формул
5. Окно активного документа
6. Окно неактивного документа
7. Панель задач **Создание книги**
8. Поле редактирования со списком **Задать вопрос**
9. Помощник по офису
10. Панель задач Microsoft Windows
11. Строка состояния
12. Кнопки документов запущенных приложений на Панели задач Microsoft Windows
13. Контекстное меню Строки состояния Microsoft Excel
14. Панель инструментов **Быстрый запуск**
15. Свернутое окно документа
16. Режим работы Microsoft Excel
17. Кнопки прокрутки ярлычков
18. Ярлычок активного рабочего листа
19. Вешки разбивки
20. Полосы прокрутки
21. Заголовки строк
22. Заголовки столбцов
23. Активная ячейка
24. Границы активной ячейки
25. Маркер автозаполнения
26. Кнопка **Выделить весь лист**

Рис. 4.1. Многодокументный интерфейс Microsoft Excel

Основное окно Microsoft Excel является типичным окном многодокументного приложения (MDI – Multiple Document Interface). В нем имеется заголовок (1), основное меню (2), панели инструментов (3), дочерние окна открытых документов (5, 6, 15) и т.д. – т.е. все, что необходимо для одновременной работы с несколькими документами. Особенностью же этого окна является наличие в нем двух элементов управления:

1. *строки (панели) формул* и
2. контекстного меню Строки состояния.

Строка формул предназначена для ускорения и облегчения ввода и редактирования формул.



В ней, слева направо, присутствуют следующие компоненты:

1. Раскрывающийся список с полем редактирования **Имя**, который может выполнять несколько функций:
   а) В режиме **Готово** здесь отображается адрес текущей ячейки.
   б) В нем можно непосредственно ввести адрес нужной ячейки, и по нажатию клавиш **Enter** выделить ее.
   в) В режиме **Укажите** (выделения) в нем отображается количество помеченных ячеек, например, 2R × 2C для области 2 × 2 ячеек.
   г) Здесь можно присвоить *имя* любому выделенному блоку ячеек.
   д) В режимах **Ввод** и **Правка** здесь отображается меню функций, с помощью которого вызывается Мастер функций Excel.
2. Кнопка со стрелкой справа от списка **Имя** позволяет раскрыть перечень поименованных объектов, если они, конечно, имеются. Щелчок по нужному имени вызывает перемещение «фокуса ввода» на требуемый объект.

3. Кнопка **Отмена** аннулирует изменения аналогично клавише **Esc**. В ячейку возвращается ее прежнее содержание, и происходит возврат в режим **Готово**.
4. Кнопка **Ввод** предназначена для подтверждения сделанных изменений аналогично клавише **Enter**. При щелчке по ней происходит выход из режима **Ввод** или **Правка** и возврат в режим **Готово**, а так же присваивание введенного значения ячейке.

   Кнопки ***Отмена*** *и* ***Ввод*** *доступны только в режимах* ***Ввод*** *и* ***Правка***.
5. Кнопка **Вставка функции** служит для вызова **Мастера функций**, облегчающего ввод функций.
6. **Поле ввода** служит, собственно, для ввода и редактирования значений ячеек.

Контекстное меню Строки состояния активизируется по щелчку правой кнопкой мыши по ней. С его помощью для выделенного диапазона ячеек можно быстро подсчитать такие параметры, как их сумма, среднее значение, максимум, минимум и т.д. Результат такого мини-вычисления также отображается в строке состояния.

6. Режимы работы

Microsoft Excel может находиться в одном из следующих *режимов*, который отображается в левой части строки состояния:

1. **Готово** – в этом состоянии возможно перемещение по таблице и доступ ко всем элементам управления.
2. **Ввод** – режим ввода данных в клетку. Большая часть меню не доступна – доступны только функции, связанные со вводом данных.
3. **Правка** – режим редактирования ранее введенных данных. Очень похож на режим **Ввод**.
4. **Укажите** – режим выбора объекта (ячейки, диапазона и т.д.) для включения его в выражение. При этом выделяемый объект ограничивается бегущей пунктирной линией.

В различных режимах одни и те же функциональные клавиши могут действовать не одинаковым образом.

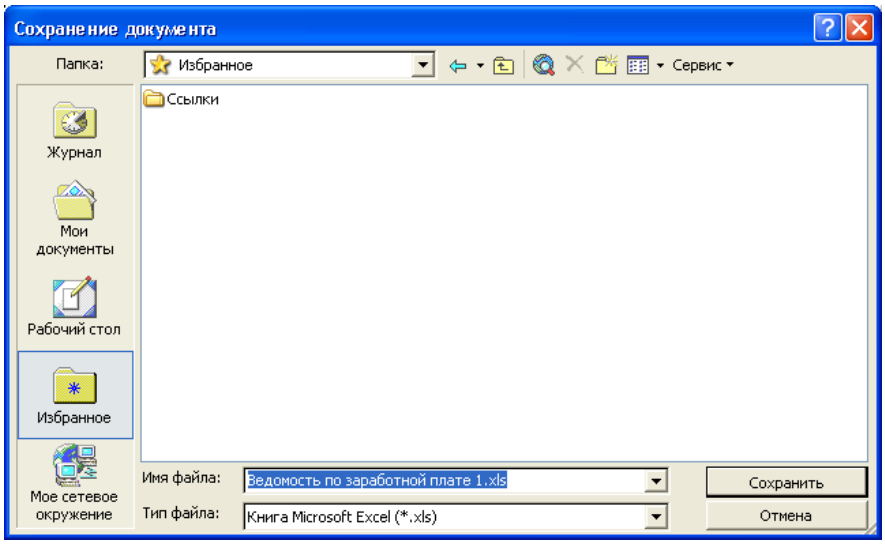
Хорошим тоном при работе с любым приложением, и Microsoft Excel в том числе, является сохранение вновь созданного документа с присвоением ему содержательного имени вместо заданного по умолчанию такого, например, как **Книга1**. Это, помимо присвоения документу осмысленного имени, позволяет также сразу иметь документ в виде дискового файла. Для сохранения документа в виде дискового файла необходимо осуществить такую последовательность шагов:

1.1. Выполнить команду **Файл ⇨ Сохранить**, или

1.2. щелкнуть по кнопке **Сохранить** на панели инструментов **Стандартная**, или

1.3. нажать сочетание клавиш **Shift** + **F12**.

2. В раскрывшемся в результате окне диалога **Сохранение документа** указать:



а) С помощью списка **Папка** – папку, в которой будет сохранена рабочая книга.

б) В поле редактирования со списком **Имя файла** – новое имя книги.

в) Нажать кнопку **Сохранить**.

В дальнейшем, по мере ввода новой информации в рабочую книгу, ее необходимо периодически сохранять, выполняя ту же команду **Файл ⇨ Сохранить**, или щелкая по кнопке **Сохранить** на панели инструментов

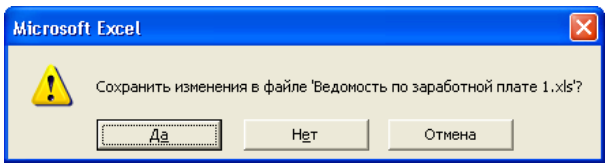
Стандартная, или нажимая сочетание клавиш **Shift** + **F12**. При этом окно диалога **Сохранение документа** больше не открывается – рабочая книга под уже имеющимся именем сохраняется «молча». Это окно диалога откроется вновь при сохранении существующей рабочей книги под новым именем, для чего необходимо выполнить команду **Файл ⇨ Сохранить как**.

7. Завершение работы

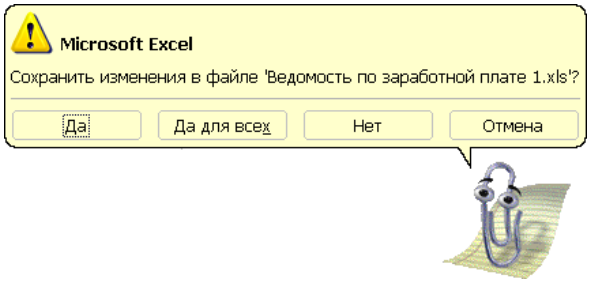
Для завершения работы Microsoft Excel необходимо:

1. выбрать команду **Файл ⇨ Выход**, или
2. нажать сочетание клавиш **Alt** + **F4**, или
3. щелкнуть на кнопке **Закрыть** в строке заголовка основного окна приложения – если был открыт только один документ. Если же документов было открыто несколько – будет закрыт только текущий документ, а Microsoft Excel продолжит работу с другими открытыми документами.

Щелчок на кнопке **Закрыть окно** на панели меню активной рабочей книги приводит к закрытию только текущего документа, без завершения работы всего приложения. Если при этом небыли сохранены последние внесенные в рабочие книгу изменения, Excel выдаст соответствующее предупреждение для каждого из таких документов.



Для сохранения изменений при закрытии рабочей книги следует щелкнуть на кнопке **Да**. При включенном режиме отображения Помощника по офису (Office Assistant) подобного рода сообщение будет отображаться Помощником.



ЛЕКЦИЯ №5

АВТОМАТИЗАЦИЯ И РАБОТА С МАКРОСАМИ

План

1. Введение
2. Программирование без программирования или запись и воспроизведение макросов
3. Просмотр макросов
4. Метаязык
5. Редактирование текста макроса
6. Копирование и перемещение макроса
7. Экспорт и импорт
8. Удаление модуля из проекта
9. Создание макроса вручную
10. Функция `MsgBox`
11. Ошибки
12. Печать
13. Настройка редактора Visual Basic

1. Введение

Каждый из нас не раз, наверное, замечал, что при использовании любой компьютерной программы для решения одной и той же задачи мы выполняем, как правило, одну и туже последовательность действий. Во время выполнения таких последовательностей команд, особенно если они длинные, мы довольно часто совершаем различного рода ошибки, не говоря уже об утомительности такой процедуры. Избежать ошибок также практически невозможно, если поручить решение такой задачи другому человеку. Однако компьютер спроектирован так, что любую последовательность действий он может выполнять сколько угодно раз, и при том – без ошибок. Поэтому создатели некоторых приложений позаботились о том, чтобы пользователь мог «записывать» («протоколировать») свои действия, а затем многократно их «воспроизводить». Использование таких «протоколов» не только повышает

скорость решения поставленных задач, но и, что более важно, – их точность и надежность.

В начале такая предварительно записанная последовательность действий, которая запускается на выполнение одной-единственной командой, называлась *макрокомандой*. В современном компьютерном лексиконе это слово сократилось до простого *макрос*, или по-английски *macro*.

 *То есть, термин макрос относится к способности приложения записывать, а затем, многократно воспроизводить последовательности выполняемых пользователем действий.*

Приемы и методы использования макросов в различных приложениях Microsoft Office несколько отличаются друг от друга. Далее, для определенности, применение макросов будем рассматривать относительно электронных таблиц Microsoft Excel. В других же приложениях они используются аналогичным образом, хотя в частностях и могут иметь некоторые особенности.

2. Программирование без программирования или запись и воспроизведение макросов

Запись макроса выполняется в четыре этапа:

1. **Создание начальных (стартовых, исходных и т.д.) условий** – т.е. приведение приложения в точно такое состояние, в котором предполагается выполнение будущего макроса.
2. **Запуск процедуры записи**. При этом необходимо *обязательно* указать:
   а) имя создаваемого макроса, и
   б) место его хранения,
   а также можно задать некоторые другие, необязательные, параметры.
3. **Выполнение действий, которые должны быть записаны в макрос**. При этом фиксируются лишь нажатия клавиш на клавиатуре и щелчки кнопок мыши, но не «маршруты» перемещения указателя мыши, и время между указанными действиями. Поэтому во время записи макроса не стоит

спешить, а выполняемые действия необходимо самым тщательным образом планировать и обдумывать.

4. **Останов записи макроса**. После останова записи макроса дальнейшие действия пользователя не фиксируются, и макрос готов к применению.

Процедуру создания макроса рассмотрим на следующем примере.

Пример 1

Предположим, что пользователю необходимо довольно часто устанавливать в некоторых ячейках рабочего листа Microsoft Excel, для привлечения к ним дополнительного внимания, шрифтом **Arial** размером **10** пунктов **полужирного** начертания. Использовать для этого один из стандартных стилей не всегда приемлемо, поскольку он может повлиять на другие параметры форматирования ячейки, такие как формат отображения числовых величин, даты, времени, выравнивание и т.д.

Исходным состоянием приложения при этом является:

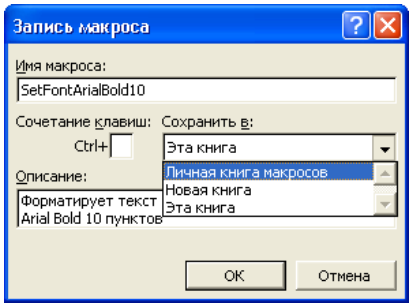
1. открытая рабочая книга Microsoft Excel,
2. на произвольном рабочем листе,
3. с выделенной одной, или несколькими ячейками.

Важность приведения приложения в исходное состояние состоит в том, что во время записи макроса все действия пользователя фиксируются самым тщательным образом – даже ошибочные, неоптимальные, и совершенно лишние. Например, если до начала записи макроса не привести Microsoft Excel в необходимое состояние, то действия по открытию рабочей книги, выделению в ней необходимого рабочего листа и нужной ячейки будут записаны в создаваемый макрос. В результате будет получен довольно-таки специфический макрос, с ограниченным диапазоном применения, – он всегда будет открывать одну и ту же книгу, один и тот же рабочий лист, и форматировать в нем одни и те же ячейки. Для того чтобы создать

универсальный макрос действия по открытию рабочей книги и выделению в ней рабочего листа и ячеек должны быть выполнены до начала записи макроса.

После приведения приложения в исходное состояние, можно приступить, непосредственно, к записи макроса, которая выполняется по такому алгоритму:

1. Выполнить команду **Сервис ⇨ Макрос ⇨ Начать запись**.
2. В раскрывшемся окне диалога **Запись макроса** с помощью его 4-х элементов управления установить параметры создаваемого макроса следующим образом:



- В поле редактирования **Имя макроса** вместо предлагаемого по умолчанию имени **Макрос*n***, где *n* – порядковый номер создаваемого макроса, указать более информативное имя, например, **SetFontArialBold10**.

  ⚠ *Длина имени макроса не должна превышать 63 символов. Оно должно начинаться с буквы, но потом может содержать и цифры. Имя макроса не должно содержать пробелов и знаков пунктуации.*

- В раскрывающемся списке **Сохранить в** указать одно из следующих возможных мест хранения записываемого макроса:
  - **Личная книга макросов**. Новый макрос будет записан в специальной книге с названием **PERSONAL.XLS**, которая открывается при каждом запуске Microsoft Excel. Такой выбор уместен, если необходимо, чтобы макрос был доступен во время любого сеанса работы, поскольку доступными являются все макросы из всех открытых рабочих книг. Если книги **PERSONAL.XLS** не

существует, Microsoft Excel создаст ее в папке своих настроек для конкретного пользователя. Как правило, это папка **\Documents and Settings\<имя пользователя>\Application Data\Microsoft\Excel\XLSTART**.

*По умолчанию книга **PERSONAL.XLS** после открытия будет скрыта, поэтому она не отображается среди открытых рабочих книг.*

- **Новая книга** – заставляет Excel создавать новую рабочую книгу и сохранять в ней записываемый макрос. При этом активной остается книга, которая была открыта до начала записи макроса, и все действия будут относиться именно к ней, а не ко вновь созданной книге.
- **Эта книга** – макрос будет записан в текущей открытой рабочей книге. Этот вариант подходит, если необходимо, чтобы макрос был доступен только при работе с текущей книгой.

*В не зависимости от того, в какой книге сохраняется макрос, он всегда будет записан на листе типа **Модуль**.*

- В поле редактирования **Сочетание клавиш** можно указать сочетание клавиш для быстрого запуска макроса. В Microsoft Excel для этого используется клавиша **Ctrl** в комбинации с какой-нибудь другой клавишей. Если в этом поле указать, например, «а», то для быстрого запуска макроса будет использоваться сочетание клавиш **Ctrl** + **a**, а если «А», то – **Ctrl** + **Shift** + **A**. Сочетание клавиш для быстрого запуска макроса следует указывать лишь в случае, если предполагается его частое использование. В противном случае не стоит понапрасну расходовать ограниченный ресурс сочетаний клавиш Microsoft Excel. При указании сочетания клавиш также необходимо соблюдать осторожность, чтобы не «перекрыть» уже зарезервированную комбинацию.
- В поле редактирования **Описание** по умолчанию в начале содержит дату записи макроса и имя пользователя, которое указано на вкладке

Общие окна диалога **Параметры**. Здесь полезно указать также, для чего предназначен макрос, и что он делает, например, Форматирует текст выделенных ячеек шрифтом Arial Bold 10 пунктов. Вся информация из этого поля помещается в начале записываемого макроса в виде *комментария*, и при его выполнении никаких действий не вызывает. Она предназначена, в основном, для пользователя, который будет применять данный макрос и, возможно, вносить в него изменения.

- ❒ Нажать кнопку **OK**.

3. В результате:
   а) в строке состояния появится слово **Запись,**
   б) активизируется панель инструментов **Остановить запись**,
   в) все дальнейшие действия пользователя будут записываться в макрос.

Панель инструментов **Остановить запись**, имеет две кнопки:

- ❒ **Остановить запись** – служит для прекращения записи макроса, и
- ❒ **Относительная ссылка** – используется для переключения между записью абсолютных и относительных адресов.

По умолчанию Microsoft Excel записывает абсолютные адреса ячеек. То есть, если во время записи макроса была активной, например, ячейка **B2**, а затем выделить ячейку справа от нее, **C2**, то при выполнении макроса каждый раз так и будет выделяться ячейка **C2**. Если же во время записи макроса был выставлен режим записи относительных адресов, то в нашем случае будет зафиксирован сдвиг ячейки на один столбик вправо без изменения номера строки, а во время его выполнения будет выделяться ячейка справа от текущей ячейки.

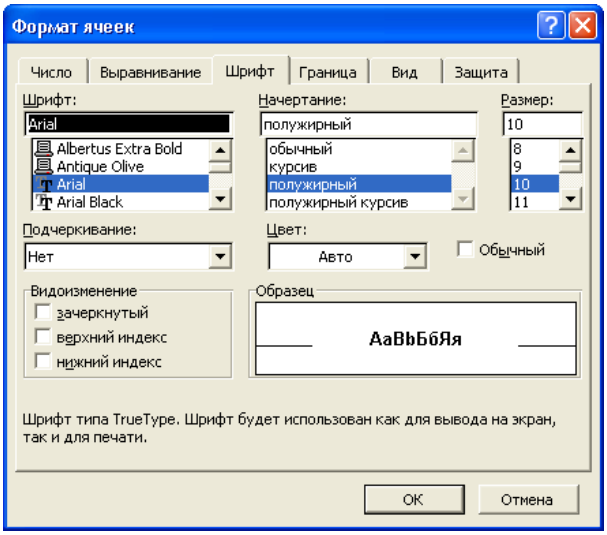
Кнопка **Относительная ссылка** – это переключатель, который имеет два положения: утопленное и приподнятое. В режиме записи абсолютных адресов она выглядит плоской, и лишь при наведении на нее указателя мыши приобретает выделенные очертания. Во время же записи

относительных ссылок она выглядит утопленной. Каждый щелчок по этой кнопке переводит ее в противоположное состояние. Переключать запись адресов с абсолютных на относительные, и обратно, можно в любой момент на протяжении записи макроса.

Для того чтобы, собственно, записать в макрос требуемые команды по изменению параметров шрифта выделенных ячеек, необходимо приступить к их выполнению, т.е.:

4. Активизировать окно диалога **Формат ячеек**:
   - ❒ по одноименной команде из контекстного меню, или
   - ❒ по команде **Формат ⇨ Ячейки**,

   в котором:



   а) Выбрать вкладку **Шрифт**, на которой:
      - ❒ в списке **Шрифт** выбрать элемент **Arial**,
      - ❒ в списке **Начертание** – **полужирный**,
      - ❒ в списке **Размер** – **10**.

      ⚠ *Если даже в выделенных ячейках эти параметры шрифта уже установлены, все равно необходимо «обновить» их, щелкнув по требуемому элементу в любом из списков.*

   б) Нажать кнопку **OK**.

В результате:

- указанные параметры шрифта будут применены к выделенным ячейкам, и
- все необходимые команды записаны в макрос.

Параметры шрифта можно установить также с помощью элементов управления на панели инструментов ***Форматирование****. Как и прежде, даже если необходимые параметры шрифта ячейки уже установлены, все равно, необходимо повторно выбрать любой из них.*

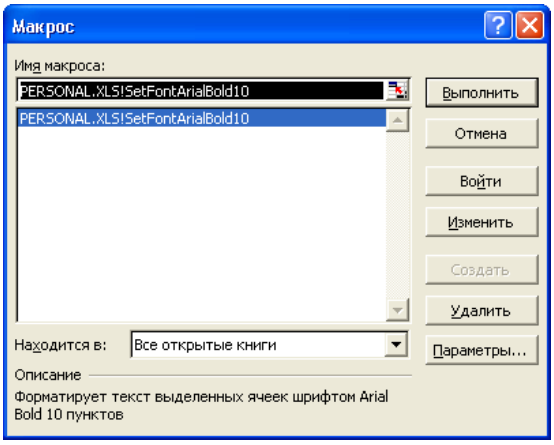
5. Завершить запись макроса:
   - ❒ щелкнув по кнопке **Остановить запись** на панели инструментов **Остановить запись**, или
   - ❒ по команде **Сервис ⇨ Макрос ⇨ Остановить запись**.

После того, как макрос создан, им можно пользоваться. Во время выполнения макроса Microsoft Excel будет исполнять все те команды, которые были в него записаны. *По умолчанию доступными для выполнения являются все макросы во всех открытых рабочих книгах. При этом книга не обязательно должна быть активной, и не обязательно должна быть видимой.*

Запустить на выполнение макрос можно:

- ❒ по нажатию сочетания клавиш, если такое было указано в окне диалога **Запись макроса**, или
- ❒ по команде **Сервис ⇨ Макрос ⇨ Макросы**.

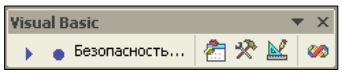
В раскрывшемся при этом окне диалога **Макрос**:



1. С помощью раскрывающегося списка **Находится в** – установить местонахождения макросов, имена которых будут присутствовать в списке **Имя макроса**, а именно:
   - **Все открытые книги** – чтобы в список попали все макросы во всех открытых рабочих книгах. Это значение установлено по умолчанию.
   - **Эта книга** – чтобы отображались только макросы из текущей рабочей книги.
   - **PERSONAL.XLS** – чтобы в список попали только макросы из личной книги макросов.
2. В списке **Имя макроса** – указать имя необходимого макроса, например, **PERSONAL.XLS!SetFontArialBold10**.
3. Нажать кнопку **Выполнить**.

В результате в выделенных ячейках шрифт изменится на **Arial полужирный 10** пунктов. Если макрос находится не в текущей книге, то перед его именем в списке **Имя макроса** указывается имя книги, в которой он был сохранен, как, например, в нашем случае – **PERSONAL.XLS!SetFontArialBold10**. Если же в списке нет нужного макроса, то предварительно необходимо открыть книгу, в которой он был сохранен.

Гораздо быстрее записывать и запускать на выполнение макросы можно с помощью кнопок на панели инструментов **Visual Basic**:



- **Записать макрос** – инициирует начало процесса записи макроса. После начала записи она превращается в кнопку **Остановить запись**.
- **Остановить запись** – останавливает процесс записи макроса, и превращается, затем, в кнопку **Запись макроса**.

- **Выполнить макрос** – инициирует начало процесса запуска макроса на выполнение.

Аналогичным образом можно создать любой макрос, записав в него любую последовательность команд. Созданный таким образом макрос можно затем запускать на выполнение сколько угодно раз. При этом, чем больше команд содержит макрос, тем больше выигрыш между «ручным» и «автоматическим» их выполнением. Существенно также, что при многократном выполнении макроса отсутствуют всякие ошибки, в то время как при «ручном» выполнении команд они, практически, неизбежны.

Макросы представляют собой наиболее мощное средство автоматизации приложений, которое позволяют их запись и выполнение! Запускать на выполнение макрос можно только в том приложении, где он был создан

3. Просмотр макросов

Все приложения, которые поддерживают запись и выполнение макросов, хранят их в стандартных файлах документов, или шаблонов. В частности, Microsoft Excel макросы хранит в файлах рабочих книг, а именно в листах типа **Модуль**. Одна рабочая книга может содержать несколько модулей, каждый из которых, в свою очередь, может содержать несколько макросов.

При записи макроса можно указать только книгу, в которой он должен быть сохранен, – текущая, новая или личная книга макросов **PERSONAL.XLS**. Модуль же для сохранения макроса Microsoft Excel выберет, или создаст, если необходимо, сам. Когда создается новый модуль для записываемого макроса, по умолчанию ему присваивается имя **Module*n***, где *n* – порядковый номер модуля в книге, создаваемого в текущем сеансе работы. Например, если в текущем сеансе работы записывается *первый* макрос и в рабочей книге (текущей или личной книге макросов) уже есть модули с именами **Module1** и **Module2**, то вновь создаваемому модулю будет присвоено имя **Module3**.

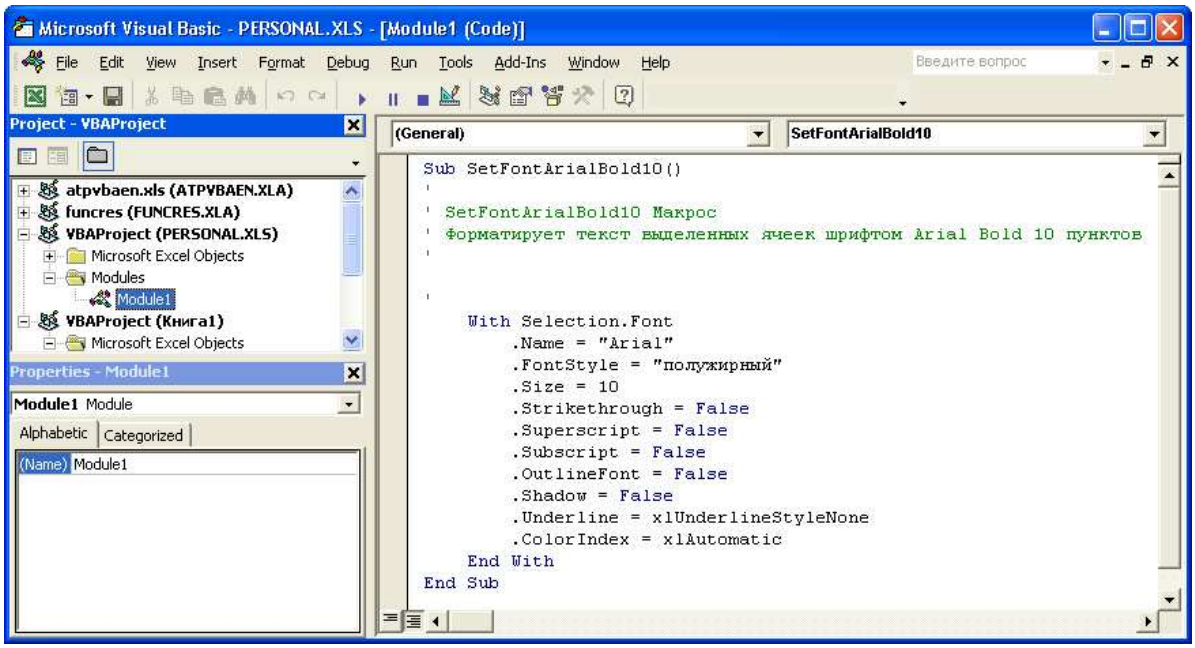
По такому же принципу строятся и имена модулей, которые

добавляются вручную с помощью команды ***Insert ⇨ Module*** *или кнопки* ***Insert Module*** *из раскрывающегося списка на панели инструментов* ***Standard*** *в редакторе Microsoft Visual Basic.*

Для создания, просмотра, редактирования и отладки макросов существует специальный инструмент, который называется Редактор Visual Basic. На выполнение его можно запустить несколькими различными способами. Наиболее простые из них это:

1. по команде **Сервис ⇨ Макрос ⇨ Редактор Visual Basic**,
2. по нажатию кнопки **Редактор Visual Basic** на панели инструментов **Visual Basic** (если она, конечно, активизирована), или
3. нажав сочетание клавиш **Alt** + **F11**.

В результате откроется основное окно приложения Microsoft Visual Basic, состоящее из заголовка, меню, панели инструментов и рабочей области – т.е., стандартное окно многодокументного (MDI) приложения Microsoft Windows. Тогда, при отображении текста записанного ранее макроса **SetFontArialBold10**, оно может выглядеть, например, следующим образом:



Рабочая область основного окна приложения Редактора Visual Basic может содержать одно, или несколько, дочерних окон следующих типов:

1. **Project** (**Проект**) – предназначено для управления проектами. *Под проектом понимается совокупность взаимосвязанных модулей, и некоторых других компонентов, хранящихся в рабочей книге, или в ее шаблоне*. Работа с этим окном практически ничем не отличается от работы с окном Проводника Microsoft Windows. В его рабочей области в виде древовидной структуры представлены все компоненты, составляющие проект. Щелчок на знаке «+» слева от некоторого элемента разворачивает данную ветвь дерева, а на знаке «-» – сворачивает. Для выделения необходимого объекта по нему необходимо один раз щелкнуть левой кнопкой мыши.

   В верхней части дочернего окна **Project**, ниже его заголовка, расположены три кнопки:

   - **View Code**, щелчок по которой вызывает отображение в окне **Code** текста выделенного элемента проекта,
   - **View Object**, которая отображает в графическом представлении выделенный в окне **Project** объект, например, лист, книгу и т.д.
   - **Toggle Folders** – это переключатель, который отображает компоненты проекта либо сгруппированными по их тапам (объекты, модули и т.д.), как представлено выше, либо – без всякой группировки, подряд.

   Окно **Project** можно открыть несколькими способами:

   - ❒ по команде **View ⇨ Project Explorer,**
   - ❒ по нажатию кнопки **Project Explorer** на панели инструментов **Standard**,
   - ❒ по нажатию сочетания клавиш **Ctrl** + **R**.

2. **Properties** (**Свойства**) отображаются свойства и значения, присвоенные каждому выбранному в окне **Project** компоненту. Причем значения свойств здесь можно изменять. Минимально, как показано на рисунке выше, свойства выбранного компонента могут состоять только из его имени. Окно содержит две вкладки:

- **Alphabetic** – свойства на которой упорядочены в алфавитном порядке их имен, и
- **Categorized** – свойства на которой сгруппированы по категориям.

Окно **Properties** можно раскрыть, если:

- ❒ выполнить команду **View ⇨ Properties Window**,
- ❒ нажать кнопку **Properties Window** на панели инструментов **Standard**,
- ❒ нажать клавишу **F4**.

3. **Code** (**Код**) – предназначено для просмотра, редактирования и создания новых кодов (или, исходных текстов) макросов. Оно может быть открыто множеством различных способов, основанные из которых будут рассмотрены далее.

   Основную, рабочую, область окна документа **Code** занимают исходные тексты макросов. В его левой нижней части расположено две кнопки, с помощью которых можно изменять режим отображения документа:

   - **Full Module View** (**Представление всего модуля**) – отображается весь модуль, со всеми входящими в него макросами. При этом каждый макрос отделяется от соседних тонкой чертой.
   - **Procedure View** (**Представление процедуры**). В этом режиме отображается только один макрос. *Макрос иногда называют также процедурой*.

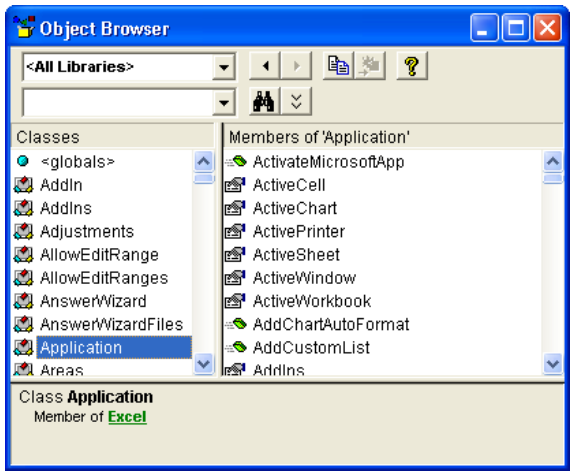
   В правой верхней части окна документа **Code** расположен раскрывающийся список **Procedure**, с помощью которого в режиме **Procedure View** можно выбирать отображаемую в нем процедуру, или быстро переходить к началу требуемой процедуры – в режиме **Full Module View**.

4. **Object Browser** (**Просмотр объектов**) – предназначено для управления всеми классами объектов, доступных в приложении. Активизируется оно по:

   - ❒ команде **View ⇨ Object Browser**,
   - ❒ нажатию кнопки **Object Browser** на панели инструментов **Standard**,

❒ нажатию клавиши **F2**,

и может выглядеть, например, следующим образом:



В левом списке **Classes** (**Классы**) перечислены все классы объектов, доступные в текущем приложении. При выборе одного из них, изменяется содержание правого списка **Members of** <**имя класса**> (**Элементы** (**члены**, **компоненты**) <**имя класса**>), содержащего элементы выбранного класса. Более детально использование этого окна будет рассмотрено далее.

Окна **Project** и **Properties**, как правило, пристыкованы к левой стороне основного окна приложения. Также может быть пристыкованым и окно **Object Browser**. Перетаскивая за заголовок каждое из этих окон, его можно пристыковать к любой из сторон родительского окна, или перевести в плавающее состояние – оставив в рабочей области, непристыковывая ни к одной из сторон. Перевести в плавающее состояние каждое из этих окон можно также, сбросив флажок пункта контекстного меню ☑ **Dockable** (**Пристыкованный**). При этом каждое из них может быть представлено только в одном единственном экземпляре. А вот окон документа **Code** может быть несколько – по одному на каждый открытый исходный модуль. Переключение между ними, а также их взаимное расположение, осуществляется с помощью команд основного меню **Window**. Границы дочерних окон в пределах рабочей области основного окна приложения можно

перемещать с помощью левой кнопки мыши, указатель которой при этом принимает вид двухсторонней стрелки. Закрытие любого из них выполняется по нажатию кнопки **Закрыть**, которая расположена в его правом верхнем углу.

Отображение текста любого, записанного ранее макроса, в окне документа **Code** можно выполнить несколькими различными способами, в зависимости от того, на сколько точно известно место его хранения. Если известно, например, что записанный ранее макрос **SetFontArialBold10** хранится в личной книге макросов **PERSONAL.XLS**, в модуле **Module1**, то вывести его в окно **Code** можно следующим образом – в окне **Project**:

1. выбрать элемент **VBAProject (PERSONAL.XLS) ⇨ Modules ⇨ Module1**,
2. щелкнуть:
   - **один раз** по кнопке **View Code**, или
   - **два раза** по выделенному элементу.

В результате в окне документа **Code** будет выведено содержимое выделенного модуля **Module1** – исходный текст единственного записанного ранее макроса **SetFontArialBold10**.

Таким способом можно отображать код *только одного* макроса, содержащегося в модуле, и то при условии, что известно имя этого модуля. Однако в большинстве случаев модуль содержит не один, а несколько макросов. При этом способ отображения макроса зависит от того:

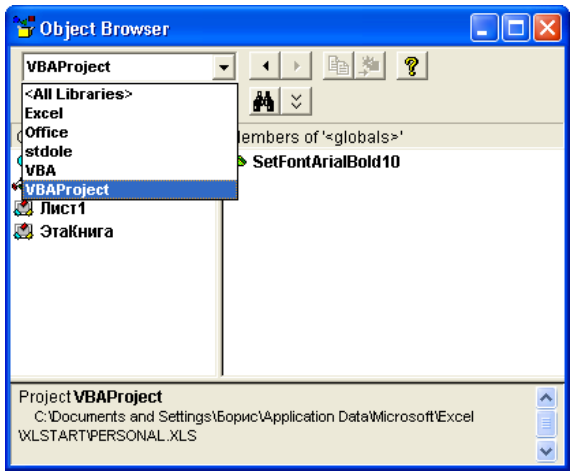
1. открыт ли в окне **Code** модуль, содержащий требуемый макрос, или
2. нет.

В первом случае необходимо:

1. просто выбрать имя этого макроса в раскрывающемся списке **Procedure**, который находится в правой верхней части окна **Code**, или
2. если модуль не слишком большой – с помощью вертикальной полосы прокрутки просто пролистать его в окне **Code**, и визуально найти требуемый макрос.

Во втором случае, т.е., если известно имя модуля, в котором хранится макрос, но сам модуль все еще не открыт в окне **Code**, открыть модуль, и сразу отобразить требуемый макрос, можно следующим образом:

1. В окне **Project** выбрать необходимый проект, например, **VBAProject(PERSONAL.XLS)**.
2. В верхнем списке **Project/Library** окна **Object Browser** выбрать необходимый компонент, например, **VBAProject**.



При этом в левом списке **Classes** (**Классы**) будут отображены все классы выбранного проекта, а в правом **Members** (**Компоненты**) – все их компоненты.

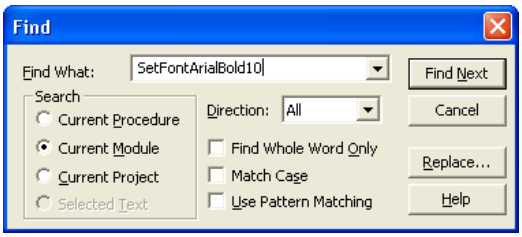
3. В правом списке **Members** (**Компоненты**) выбрать необходимый макрос, например, **SetFontArialBold10**:
   - ❒ выполнив на нем двойной щелчок, или
   1. выделив его одинарным щелчком, и
   2. нажав кнопку **View Definition** (**Просмотр определения**).

Если же положение макроса известно лишь с точностью до проекта, то его отображение выполняется следующим образом:

1. В окне **Project** – выбрать имя проекта, содержащего требуемый макрос.
2. В указанном проекте выполнить поиск макроса одним из приведенных далее способов.

Первый из них осуществляется с помощью окно диалога **Find** (**Искать**) и заключается в следующем:

1. При *активном* окне документа **Code**, открыть диалоговое окно поиска **Find** одним из способов:
   - ❒ выполнив команду **Edit ⇨ Find**, или
   - ❒ нажав кнопку **Find** на панели инструментов **Standard**, или
   - ❒ нажав сочетание клавиш **Ctrl** + **F**.
2. В окна диалога **Find**:



   - ❒ в раскрывающемся списке **Find What** (**Что искать**) – указать имя требуемого макроса, например, **SetFontArialBold10**,
   - ❒ с помощью переключателя **Search** – указать диапазон поиска:

     Current Procedure – в пределах текущей процедуры (макроса),

     Current Module – в пределах текущего модуля,

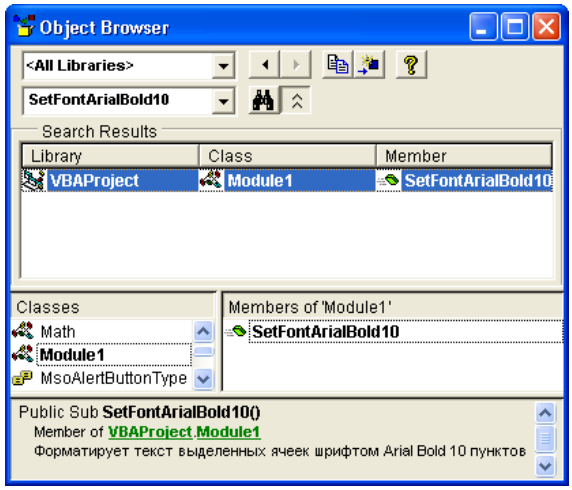
     Current Project – в пределах текущего проекта,

     Selected Text – в пределах выделенного текста.
   - ❒ щелкнуть по кнопке **Find Next** (**Искать далее**).

Второй способ выполняется с помощью дочернего окна **Object Browser**, и состоит в следующем:

1. В его верхнем списке **Project/Library** (**Проект/Библиотека**) – выбрать элемент **<All libraries>** (**<Все библиотеки>**).
2. В нижнем списке **Search Text** (**Текст поиска**) – ввести имя макроса, например, **SetFontArialBold10**.
3. Щелкнуть по кнопку **Search** (**Поиск**).

4. В раскрывшемся списке **Search Results** (**Результаты поиска**) – выбрать необходимый макрос, например, **SetFontArialBold10**:
   - ❒ выполнив на нем двойной щелчок, или
   1. выделив его одинарным щелчком, и
   2. нажав кнопку **View Definition** (**Просмотр определения**).



В результате в окне документа **Code** будет выведен код требуемого макроса. Однажды введенное в список **Search Text**, или **Find What**, имя макроса остается там до конца текущего сеанса, и в случае повторного поиска его вводить больше не нужно – можно просто указать его как элемент раскрывающегося списка.

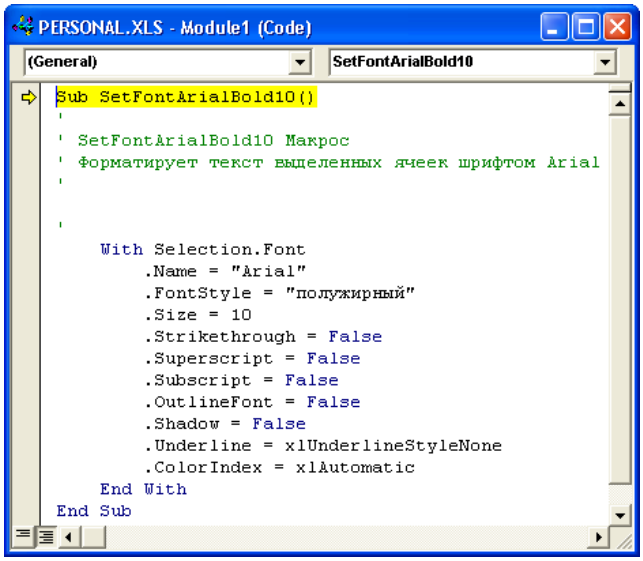
Если после щелчка на кнопке **Search** (**Поиск**) поиск не дал результата – в указанном проекте макрос не был обнаружен – продолжить его поиск можно, не закрывая списка **Search Results** (**Результаты поиска**), выбирая в окне **Project** другие проекты. В окне **Project** при этом отображаются только открытые в данный момент проекты. Если же нужного проекта там нет, то необходимо из редактора Visual Basic перейти в Microsoft Excel, например:

- ❒ нажав кнопку **View Microsoft Excel** на панели инструментов **Standard**, или
- ❒ по команде **View ⇨ Microsoft Excel**, или
- ❒ нажав сочетание клавиш **Alt** + **F11**,

и там уже открыть необходимую книгу.

Если же точно не известно, в каком именно проекте находится макрос, то необходимо:

1. В Microsoft Excel открыть все рабочие книги, в одной из которых предполагается его наличие.
2. Активизировать окно диалога **Макрос**:
   - ❒ по команде **Сервис ⇨ Макрос ⇨ Макросы**, или
   - ❒ нажав сочетание клавиш **Alt** + **F8**.
3. В списке **Имя макроса** – указать необходимый макрос. При этом если макрос находится в одном из модулей активной рабочей книги, в списке он будет присутствовать как элемент, состоящий только из имени. Если же в неактивной, например, личной книге макросов **PERSONAL.XLS**, – в списке перед его именем будет указано и имя книги, например, **PERSONAL.XLS!SetFontArialBold10**.
4. Нажать кнопку:
   - 1.1. **Изменить** – если макрос находится в любой из открытых книге, *кроме скрытой личной книги макросов* ***PERSONAL.XLS***, или
   - 1.2. **Войти** – если макрос находится в личной книге макросов **PERSONAL.XLS**. При этом макрос запустится на выполнение в пошаговом режиме, и стрелкой на поле слева от текста будет отмечена первая строка, подлежащая выполнению.



2. **Reset** (**Сброс**) на панели инструментов **Standard** редактора Visual Basic, чтобы прекратить выполнение макроса и перейти в режим его редактирования.

Код макроса **SetFontArialBold10**, в не зависимости от того, каким способом он был отображен в окне **Code**, приведен в следующем листинге:

```
1:  Sub SetFontArialBold10()
2:  '
3:  ' SetFontArialBold10 Макрос
4:  ' Форматирует текст выделенных ячеек шрифтом Arial Bold 10 пунктов
5:  '
6:
7:  '
8:      With Selection.Font
9:          .Name = "Arial"
10:         .FontStyle = "полужирный"
11:         .Size = 10
12:         .Strikethrough = False
13:         .Superscript = False
14:         .Subscript = False
15:         .OutlineFont = False
16:         .Shadow = False
17:         .Underline = xlUnderlineStyleNone
18:         .ColorIndex = xlAutomatic
19:     End With
20: End Sub
```

Он несколько отличается от текста в окне **Code** – в начале каждой строки в нем добавлен ее номер. В листингах этого издания такая нумерация выполнена

исключительно для удобства ссылок на отдельные строки кода, с целью их объяснения.

Но, каким бы способом не был отображен исходный текст макроса, первый раз он все равно выглядит, наверное, незнакомым и совершенно непонятным. Так насладитесь же в последний раз своим невежеством, ибо скоро от него не останется и следа!

В процессе записи макроса команды, выполняемые пользователем, трансформируются в исходный текст языка VBA – Visual Basic for Applications. Язык VBA является языком программирования, который:

- с одной стороны является универсальным языком программирования, а
- с другой стороны – поддерживает средства доступа ко всем, специфическим, возможностям того приложения, из которого он был запущен.

Язык программирования VBA является результатом дальнейшего развития такого широко распространенного языка программирования, как BASIC – Beginner's All Purpose Symbolic Instruction Code – символьный язык программирования общего назначения для начинающих. Несмотря на то, что VBA для каждого из приложений, которое его поддерживает (Microsoft Excel, Microsoft Word и т.д.), имеет свой, особенный, набор возможностей, общая основа у него остается неизменной. То есть, освоив однажды VBA, например, для Microsoft Excel, использование его, затем, например, для автоматизации Microsoft Word не составит особого труда.

Создать исходный код на VBA можно двумя различными способами:

1. автоматически – записывая выполняемые пользователем команды, и трансформируя их в инструкции VBA, или
2. вручную – вводя необходимый текст с клавиатуры.

Текст, созданный первым способом, традиционно называется *макросом*, тогда как вторым – *процедурой*. После того, как исходный код был записан в модуль, уже нельзя определить, как он туда попал – вручную, или автоматически.

Но между этими двумя способами создания исходного кода VBA есть некоторые, иногда довольно существенные, различия. При автоматическом

способе исходный код «генерируется» очень быстро, и притом без ошибок. Но программа получается довольно-таки «грубой», поскольку может выполнять только *последовательный* набор предписанных действий, и никак не может реагировать на изменяющиеся во время ее выполнения условия. При ручном способе создания исходного текста можно учесть практически все нюансы и условия выполнения будущей программы. Но создание и отладка такого кода требует гораздо больших временных и интеллектуальных затрат. На практике же оба эти способа часто используют совместно, т.е.:

1. в начале – автоматически записывают фрагменты будущей программы, а
2. затем – ее корректируют вручную, дополняя необходимой логикой и гибкостью.

Исходный текст макроса (или процедуры) состоит из *инструкций*, которые образуют:

1. *заголовок*,
2. *тело*, и
3. *окончание*.

Заголовок определяет имя процедуры, и ее начало в тексте модуля, а тело – выполняемые действия. Заголовок, в свою очередь, состоит из:

а) *ключевого слова* ***Sub*** (от Subroutine – подпрограмма),

б) имени процедуры, которое формируется с учетом изложенных выше правил для имен автоматически записываемых макросов, и

в) пары круглых скобок – ().

После тела макроса всегда следует пара ключевых слов ***End Sub***, которые говорят о том, что макрос закончился.

Ключевые (служебные, зарезервированные) слова VBA предназначены для внутренних нужд языка, и их нельзя использовать ни для каких других целей, кроме – специально оговоренных!

В тексте издания они выделены ***моноширинным полужирным курсивом***. Тогда, структура процедуры, в общем виде, будет выглядеть следующим образом:

```
Sub <имя>()
    [<инструкции>]
End Sub.
```

4. Метаязык

При рассмотрении любого языка программирования обычно выделяют:

1. *синтаксис* и
2. *семантику*.

Синтаксис, или грамматика, языка программирования (как и любого естественного языка) – это набор правил, точно определяющих множество его правильных конструкций. Семантика языка – это набор правил, точно описывающих выполнение синтаксически правильных конструкций на реальном или абстрактном компьютере. Фактически правила семантики описывают необходимые действия по выполнению каждой инструкции программы.

Для описания синтаксиса и семантики языков программирования используются специальные языки – *метаязыки* («надязыки»). В данном издании в качестве метаязыка будет использована следующая система обозначений:

1. Ключевые слова языка программирования выделены ***полужирным шрифтом***, и в тексте программы записываются так, как приведено здесь.
2. Текст, заключенный в символы «<>», необходимо заменить правильной конструкцией языка.
3. Конструкции, заключенные в символы «[]», являются не обязательными, и в некоторых случаях могут быть опущены.
4. Символ «|» служит для разделения нескольких возможных альтернатив.
5. Сами же символы «<>», «[]» и «|» относятся к метаязыку, а не к описываемому языку программирования, и при записи текста программы опускаются.

В листинге приведенного выше макроса **SetFontArialBold10**:

- строка 1 – это заголовок,
- строки 2÷19 – тело, а
- строка 20 – завершает его.

Строки 2÷7 в начале макроса, которые начинаются символом апострофа (') являются *комментариями*. Они были перенесены сюда из поля **Описание** окна диалога **Запись макроса**, которые пользователь занес туда во время записи макроса. В процессе выполнения макроса комментарии никаких действий не вызывают, а служат лишь пояснением тому, кто будет его читать.

Инструкции языка программирования VBA, в какой-то мере, эквивалентны предложениям естественного языка – каждая из них выражает некоторую законченную мысль (или действие). Всякая инструкция:

1. по умолчанию – занимает отдельную строку кода, но
2. в одной строке может быть записано несколько инструкций – разделяются они при этом символом двоеточия (:), а также
3. отдельная инструкция может занимать несколько строк, каждая из которых, за исключением последней, должна обязательно заканчиваться парой символами – *пробелом* и *подчеркивания* (_). Разделение инструкции на части служит для ее лучшего восприятия во время чтения. При этом если длинная инструкция не будет разделена на отдельные строки, она может занять в окне так много места, что для того чтобы добраться до ее конца, придется очень долго его прокручивать.

Инструкции VBA, составляющие тело макроса, соответствуют тем действиям, которые выполнял пользователь во время его записи. Так, например, строка

```
.Size = 10
```

соответствует установке размера шрифта в 10 пунктов. Во время же выполнения макроса инструкции выполняются строго по порядку – от начала, и

до конца – и каждая из них вызывает вполне определенное действие. Комментарии при этом, естественно, пропускаются. Комментарий в VBA может занимать не только строку целиком, как в приведенном выше листинге, но и ее часть – от места его начала в строке (знака апострофа), и до ее конца.

Код представленного выше макроса отформатирован так, что различные его строки записаны с различной величиной отступа от его левого края. Это подчеркивает его логическую структуру, а также способствует улучшению его визуального восприятия. Все тело макроса записано с отступом относительно инструкций ***Sub*** и ***End Sub***, что помогает его вычленению из остального текста. В строках 9÷18 отступ еще больше. Это сделано с целью выделения строк, заключенных между инструкциями ***With*** и ***End With***.

Программа записи макросов автоматически делает такие отступы для того, чтобы пользователю легче было его читать. Когда текст процедуры записывается вручную, также следует делать такие отступы для выделения его различных логических фрагментов. При этом не существует никаких определенных правил, относительно того, как делать отступы, и делать ли их вообще. Приведенный выше макрос будет работать точно также, если все его строки выровнять по левому краю. Однако некоторые правила все-таки есть, и сформулировать их можно следующим образом:

1. *отступы способствуют улучшению читабельности кода программы, и*
2. *при этом необходимо всегда придерживаться один раз установленного порядка их применения.*

Если при работе с макросами используется цветной монитор, то редактор VBA в окне **Code** их текст расцвечивает следующим образом:

1. зеленым цветом выделяются комментарии,
2. синим цветом выделяются ключевые слова, такие как ***Sub***, ***End Sub*** и другие, и
3. черным цветом – все остальное.

Это служит дополнительным средством структурирования текста программы.

5. Редактирование текста макроса

Редактирование текста макроса в окне документа **Code** основано на тех же методах и приемах, что и редактирование любого другого текстового документа. Все делается точно также как и в программах Блокнот, WordPad или Microsoft Word. В качестве примера отредактируем текст записанного ранее макроса **SetFontArialBold10** так, чтобы непомещающийся в одной строке 4 комментарий разделить на две строки. Для этого необходимо:

1. В строке 4 после слова ячеек нажать клавишу **Enter**. При этом строка 4 «разломится» две строки – 4 и 5.
2. В первой колонке строки 5 ввести признак комментария – символ апострофа (**'**):
   - ❒ вручную с клавиатуры, или
   - а) выделив эту строку, и
   - б) нажав кнопку  **Comment Block** (**«Закомментировать» блок**) на панели инструментов **Edit**.
3. Удалить пустую строку 7.

В результате будет получен следующий код:

```
1:   Sub SetFontArialBold10()
2:   '
3:   ' SetFontArialBold10 Макрос
4:   ' Форматирует текст выделенных ячеек
5:   ' шрифтом Arial Bold 10 пунктов
6:   '
7:   '
8:       With Selection.Font
9:           .Name = "Arial"
10:          .FontStyle = "полужирный"
11:          .Size = 10
12:          .Strikethrough = False
13:          .Superscript = False
14:          .Subscript = False
15:          .OutlineFont = False
16:          .Shadow = False
17:          .Underline = xlUnderlineStyleNone
18:          .ColorIndex = xlAutomatic
19:      End With
20:  End Sub
```

6. Копирование и перемещение макроса

Копирование и перемещение макроса выполняется точно так же, как и любого другого фрагмента текста в текстовом редакторе – через буфер обмена. Для этого необходимо:

1. Выделить подлежащий копированию, или перемещению, макрос, *включая строки с ключевыми словами* ***Sub*** *и* ***End Sub***.
2. С помощью соответствующих команд меню, или кнопок панели инструментов, или сочетания клавиш, скопировать, или переместить, в буфер обмена выделенный фрагмент кода.
3. С помощью текстового курсора указать в выходном модуле то место, куда будет вставлен код из буфера обмена.
4. Любым известным способом вставить в позиции курсора содержимое буфера обмена.

⚠ *Редактор Visual Basic не контролирует того, чтобы в одном модуле не получилось двух, и более макросов, с одинаковыми именами – ответственность за это возлагается на пользователя. Такая ошибка будет обнаружена лишь во время попытки запуска на выполнение этого макроса.* Перемещать и копировать фрагменты кода можно также при помощи мыши – методом Перетащить и Отпустить. Техника выполнения этого приема та же, что и обычно.

Резервную копию макроса рекомендуется создавать всегда:

- ❒ перед тем как вносить в него много изменений, или
- ❒ если макрос большой и выполняет сложные действия,
- ❒ и т.д.

Сделанная копия является страховкой, на случай если из выполненных переделок ничего не выйдет, или выйдет не то, что ожидалось. В этом случае необходимо будет просто осуществить «откат» к предыдущей версии.

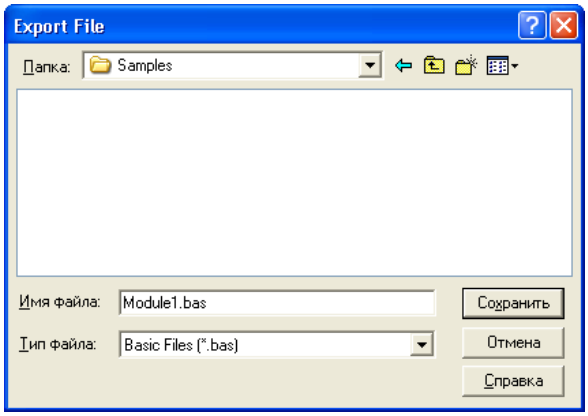
7. Экспорт и импорт

Экспорт и импорт позволяют осуществить обмен *модулями* между различными проектами в рамках одного приложения, или между различными

приложениями, а также для создания архива программ в файлах текстового формата. Кроме того, это позволяет осуществить преобразование модуля в формат Visual Basic версии 6.0, и старше (другие, более ранние версии, просто не проверялись). При этом под экспортом понимается запись модуль во внешний текстовый файл, а под импортом – вставка такого файла в любой VBA-, или Visual Basic-проект.

Для того чтобы экспортировать модуль в текстовый файл, необходимо:

1. В окне **Project** выделить подлежащий экспорту модуль.
2. Выполнить команду **File ⇨ Export File**.
3. В раскрывшемся при этом окне диалога **Export File**, аналогичном стандартному окну диалога **Сохранение документа**:



а) В раскрывающемся списке **Папка** – указать папку, в которой будет записан экспортируемый модуль.

б) В поле **Имя файла** – указать имя текстового файла, под которым он будет сохранен.

в) В раскрывающемся списке **Тип файла** – обязательно выбрать значение **Basic Files (*.bas)**.

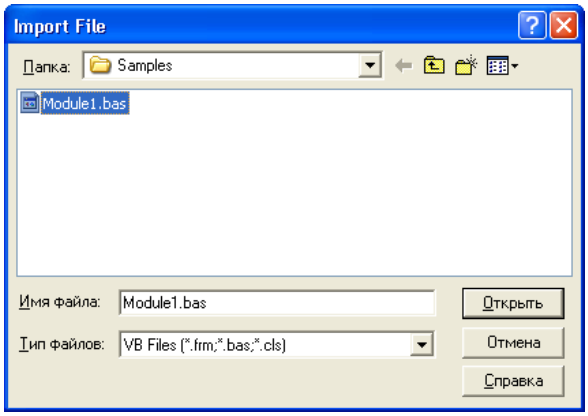
г) Щелкнуть по кнопке **Сохранить**.

В результате будет создан текстовый файл с расширением **.bas**, в который, помимо основного кода, будет добавлена некоторая служебная информация, которая используется для того, чтобы этот файл, затем, можно было

импортировать в другой проект. При желании содержимое этого файла можно просмотреть с помощью любого текстового редактора.

Импорт, или присоединение, экспортированного ранее файла типа ***.bas** к VBA-проекту выполняется следующим образом:

1. В окне **Project** выделить проект, в который необходимо импортировать содержимое файла типа ***.bas**.
2. Выполнить команду **File ⇨ Import File**.
3. В раскрывшемся окне диалога **Import File**, аналогичном стандартному окну диалога **Открытие документа**:



   а) В раскрывающемся списке **Папка** – выбрать папку, в которой следует искать требуемый файл.

   б) В списке **Тип файла** – выбрать значение **VB Files (*.frm, *.bas, *.cls)**.

   в) Выбрать требуемый файл:

   - ❒ дважды щелкнув по нему кнопкой мыши, или

   1. один раз щелкнув по нему, и
   2. один раз щелкнув по кнопке **Открыть**.

В результате к проекту будет добавлен модуль. При этом если в проекте уже имеется модуль с таким же именем, как и у импортируемого, редактор Visual Basic добавит единицу к имени нового модуля.

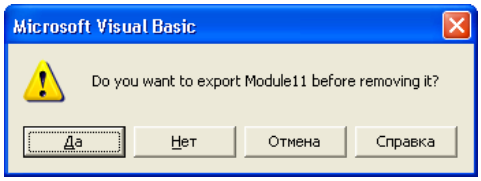
8. Удаление модуля из проекта

Удаление модуля из проекта выполняется, если надобность в нем отпала. При этом причины могут быть самые разные:

- ❒ макросы, записанные в модуле, устарели и у них появились новые, более совершенные версии;
- ❒ в модуль были записаны «экспериментальные» версии макросов, созданные лишь для проверки некоторых гипотез;
- ❒ и т.д.

Процедура удаления ненужного более модуля из проекта выполняется по такому алгоритму:

1. В окне **Project** выделить подлежащий удалению модуль.
2. Выполнить команду **Remove <имя модуля>** (**Удалить <имя модуля>**) из контекстного или основного меню **File**.
3. В раскрывшемся окне сообщения будет предложено экспортировать удаляемый модуль.



При этом если ответить **Да**, то будет запущена процедура экспорта модуля, если же **Нет** – модуль будет удален безвозвратно. Поэтому, прежде чем удалять модуль, необходимо окончательно убедиться в том, что данный модуль больше никогда не понадобится. И если существует хоть малейшее сомнение – модуль лучше все-таки экспортировать. Аналогичным образом можно удалять не только модули, но и другие компоненты проекта.

9. Создание макроса вручную

Создание макроса (или процедуры) вручную можно выполнить, записав его:

1. в существующий модуль, или

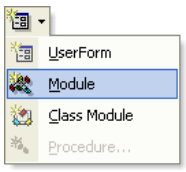
2. во вновь созданный.

Новый модуль создается в случаях, если:

- ❒ книга, в которой необходимо создать процедуру не содержит ни одного модуля;
- ❒ существующие модули переполнены (каждый из них не может содержать более 4000 строк кода);
- ❒ необходимо поместить новую процедуру в отдельный модуль;
- ❒ и т.д.

Для добавления нового модуля к проекту необходимо:

1. В окне **Project** выделить проект, подлежащий обновлению.
2. Вставить модуль:
   - ❒ выполнив команду **Insert ⇨ Module** из контекстного или основного меню, или
   - ❒ указать значение **Module** из раскрывающегося списка вставки на панели инструментов **Standard**.



Вновь созданному модулю Visual Basic присвоит имя в соответствии с правилами, приведенными выше, т.е. **Module**, и порядковый номер. Такое имя мало, о чем говорит. Поэтому после создания его лучше сразу переименовать, присвоив более содержательное имя, чтобы по нему можно было определить назначение процедур, содержащихся в этом модуле. А выполняется это следующим образом:

1. В окне **Project** выделить модуль, подлежащий переименованию.
2. В окне **Properties** в текстовом поле **Name** ввести новое имя модуля. И как только курсор покинет это поле, модуль будет переименован.

При добавлении новой процедуры в модуль, первое, что необходимо сделать, – это определить в нем место, куда будет вставлена новая процедура. Для нового модуля никаких особых вариантов не существует – добавляемая процедура всегда записывается с начала пустого модуля. Для существующего модуля, ужа содержащего процедуры, новая процедура может быть записана в любом месте (в начале, в средине или в конце), кроме текста уже существующих процедур – *процедуры не могут быть вложенными друг в друга как матрешки*. Однако, в большинстве случаев, новые процедуры, как правило, записываются в конец модуля.

Создание новой процедуры вручную рассмотрим на примере, с которого традиционно начинается знакомство с новым языком программирования, – программа в момент своего «рождения» приветствует весь Мир словами – «Здравствуй, Мир!». Для этого в окне **Code** необходимо ввести код процедуры, листинг которой представлен далее:

```
1.  Sub Hello()
2.     MsgBox "Здравствуй, Мир!"
3.  End Sub
```

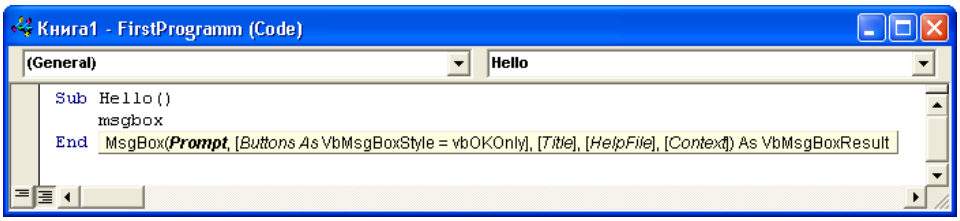
 *Номера строк при этом вводить, естественно, не надо – они служат, как было отмечено ранее, лишь для удобства ссылки на них в этом издании.*

Редактор Visual Basic во многом помогает в процессе создания программ. Так, в первой строке объявления процедуры необходимо ввести только ключевое слова ***Sub*** и имя процедуры, а когда курсор ее покинет, автоматически будут добавлены:

1. круглые скобки после имени процедуры, и
2. строка завершения процедуры – ***End Sub***.

Ключевые слова, такие, например, как ***Sub***, при этом также необязательно вводить с большой буквы – редактор Visual Basic это сделает сам. Наоборот, их лучше вводить строчными буквами, поскольку если после того, как курсор покинет текущую строку, ключевые слова будут выведены с

большой буквы, – это может служить дополнительным признаком того, что все они были введены правильно. Автоматическое добавление строки завершения процедуры ***End Sub*** также способствует уменьшению вероятности появления в ней ошибок – заголовок и завершение процедуры сформированы, практически, автоматически – осталось записать только ее тело!? Но и здесь редактор Visual Basic поможет – после ввода ключевого слова ***MsgBox*** и пробела после него появляется вспомогательное окно, в котором отображается *прототип функции*:



Имя аргумента, который необходимо вводить сейчас, выделено в нем ***полужирным*** начертанием. Это средство называется **Auto Quick Info** (**Автоматическая быстрая информация**), а закрывается окно когда:

- ❒ завершен ввод функции (и всех ее параметров), или
- ❒ курсор покидает пределы строки, или
- ❒ нажата клавиша **Esc**.

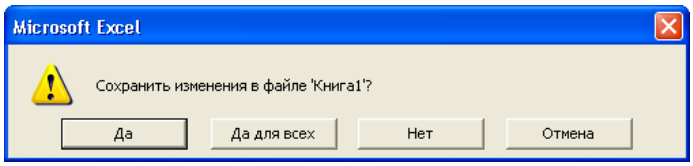
Даже в такой маленькой процедуре, чтобы подчеркнуть ее структуру, тело записано с отступом.

Для того чтобы проделанная работа не пропала, созданную процедуру необходимо сохранить любым из способов:

- ❒ выполнить команду **File ⇨ Save**,
- ❒ нажать кнопку **Save** на панели инструментов **Standard**,
- ❒ нажать сочетание клавиш **Ctrl** + **S**.

При этом она будет сохранена в текущей рабочей книге. Если же этого не сделать, то при завершении текущего сеанса Microsoft Excel будет выдано окно

сообщения для каждой из рабочих книг, в которую были внесены изменения, но она не была сохранена.



Подтвердив необходимость сохранения, таким образом, можно записать на диск внесенные в рабочие книги изменения, в том числе и созданные процедуры.

Далее более подробно рассмотрим нашу первую процедуру:

Строка 1. это инструкция объявления процедуры – ее заголовок. В ней указывается, что процедура является подпрограммой с именем **Hello**, и служит она, чтобы указать начало процедуры в тексте модуля.

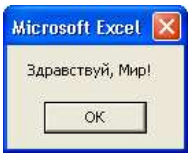
Строка 2. единственная инструкция, составляющая тело процедуры. Она состоит из обращения (или вызова) функции ***MsgBox***, которая выводит в стандартном окне сообщения содержимое своего первого аргумента. Более подробно функция ***MsgBox*** будет рассмотрена далее.

Строка 3. инструкция завершения процедуры ***End Sub***. Встретив ее в тексте модуля, Visual Basic завершает выполнение процедуры.

После того, как создана новая, или отредактирована существующая процедура, необходимо проверить, как она работает. Это можно выполнить следующим образом:

1. Установить курсор в пределах процедуры, включая строки ее заголовка и завершения.
2. Запустить ее на выполнение одним из способов:
   а) выполнить команду **Run ⇨ Run Sub/UserForm**,
   б) нажать кнопку **Run Sub/UserForm** на панели инструментов **Standard**,
   в) нажав клавишу **F5**.

В результате выполнения процедуры **Hello** будет выведено следующее окно сообщения:



Если же в момент запуска процедуры курсор находится в не ее пределов, Visual Basic не сможет определить, какую именно процедуру необходимо запускать, и он выводит окно диалога **Macros**, аналогичное рассмотренному ранее окну диалога **Макрос**, с помощью которого можно запустить любую процедуру – и созданную вручную, и записанную автоматически.

10. Функция *MsgBox*

Функция ***MsgBox***, составляющая тело процедуры **Hello**, осуществляет вывод информации пользователю в стандартном окне сообщения. Информация, которую программа выводит на экран, принтер, в файл и, вообще, на любое внешнее устройство компьютера, называется *выводом*. Функция ***MsgBox*** – это наиболее распространенный способ вывода информации из процедуры на экран дисплея. Она является *стандартной* (или *встроенной*) функция VBA. Инструкция процедуры **Hello**, в которой встречается имя функции ***MsgBox***, осуществляет ее вызов, или обращение к ней, с «просьбой» осуществить вывод сообщения.

Однако, такую «просьбу» функция ***MsgBox()*** непосредственно выполнить не может – она должна знать, что именно нужно вывести. Для этого при обращении к ней передается строка данных заключенная в двойные кавычки – `"Здравствуй, Мир!"`. Такую дополнительную информацию, передаваемую в функцию при ее вызове, называют *аргументами* (или *параметрами*). Первый параметр функции ***MsgBox*** называется ***Prompt*** (***Приглашение***), и представляет собой текст, который необходимо вывести. А для того, чтобы указать VBA, что эта строка является данными, которые

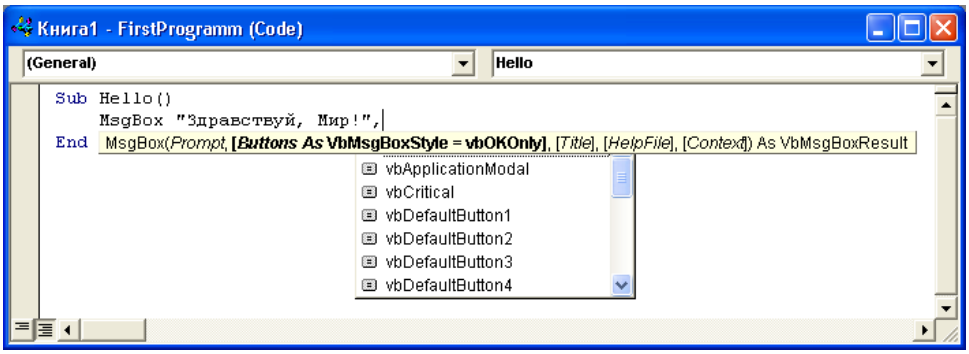
необходимо обработать (в данном конкретном случае, вывести на экран), а не ключевым словом языка, или другим компонентом программ, ее необходимо заключить в кавычки.

В заголовке окна сообщения, выводимого функцией ***MsgBox***, по умолчанию отображается название того приложения, из которого оно было активизировано – в данном случае Microsoft Excel. Однако, это не всегда приемлемо, и информативно. Для того чтобы изменить заголовок окна сообщения, необходимо откорректировать процедуру **Hello** следующим образом:

```
1:   Sub Hello()
2:      MsgBox "Здравствуй, Мир!", , "Окно приветствия"
3:   End Sub
```

В ней изменения коснулись только строки 2. Она хоть и выполняет тоже, что и ранее действие, – выводит сообщение, но выглядит несколько по-иному – теперь она содержит три аргумента, разделенных запятыми. Первый аргумент функции ***MsgBox*** остался тот же, что и ранее – текст выводимого сообщения.

При редактировании инструкции вызова функции ***MsgBox***, как и прежде, открывается вспомогательное окно сведений, в котором отображается ее прототип. Но если по какой-либо причине оно не будет открыто автоматически, вручную его можно открыть по нажатию комбинации клавиш **Ctrl** + **I**.

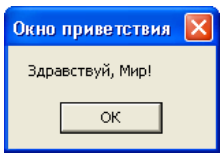


Для перехода ко вводу второго параметра необходимо после первого параметра поставить запятую. При этом во вспомогательном окне сведений

прототип функции изменится – первый параметр будет выведен шрифтом нормального начертания, а второй – ***полужирным*** – признак того, что он является текущим. Второй параметр называется **`Buttons`** (**`Кнопки`**) и служит для того, чтобы указать, сколько и какие кнопки должны присутствовать в окне сообщения. После ввода пробела вслед за запятой раскроется список, из которого можно выбрать одно из допустимых значений второго параметра. Но поскольку второй параметр необязателен, и в прототипе функции он выведен в квадратных скобках, его можно опустить. Признаком того, что очередной параметр в списке параметров опущен, является две подряд идущие запятые. *Даже если очередной параметр опущен, запятая все равно ставится!* Значением по умолчанию для второго параметра является **`vbOKOnly`**, которое соответствует единственной кнопке **OK**. После ввода запятой вместо второго опущенного параметра содержимое вспомогательного окна сведений снова изменится – полужирным начертанием теперь будет выведен третий параметр.

Третий параметр функции **`MsgBox`** называется **`Title`** (**`Заголовок`**) и служит для указания заголовка окна сообщения. Поскольку этот параметр – текстовая строка, которая должна отображаться в заголовке окна сообщения, то в инструкции обращения к функции его, как и первый параметр, необходимо заключать в двойные кавычки. Если эти кавычки опустить, то VBA выдаст соответствующее сообщение об ошибке. Поскольку этот параметр последний в списке, и другие указываться не будут, запятые больше ставить не нужно – *вместо последних опущенных параметров запятые не ставятся*.

Тогда в результате выполнения новой, откорректированной, версии нашей процедуры будет выдано следующее окно сообщений:



11. Ошибки

Ошибки – это постоянные и неизменные спутницы процесса создания программ. Без них не обходится ни одно создание программы, и в этом Вы сами скоро убедитесь. При этом VBA обнаруживает большинство из них и выдает необходимые сообщения. Все ошибки, которые могут встретиться в программе, делятся на две категории:

2. *синтаксические*, и
3. *семантические*, или *времени выполнения* (*run-time error*).

Наиболее распространенной категорией ошибок являются синтаксические ошибки, связанные с нарушением грамматики языка программирования. Они информируют о таких неприятностях, как:

- ❒ пропущенные запятые,
- ❒ недостающие кавычки,
- ❒ отсутствующие обязательные аргументы функции,
- ❒ и т.д.

После того, как курсор покинет пределы *откорректированной* строки программы, выполняется два, подряд идущих, действия:

1. *синтаксический анализ*, и
2. *трансляция*.

Под синтаксическим анализом понимается процесс, при котором исходный текст программы раскладывается на составляющие элементы (*лексемы*), такие как:

- ❒ ключевые слова,
- ❒ элементы данных,
- ❒ разделители,
- ❒ и т.д.

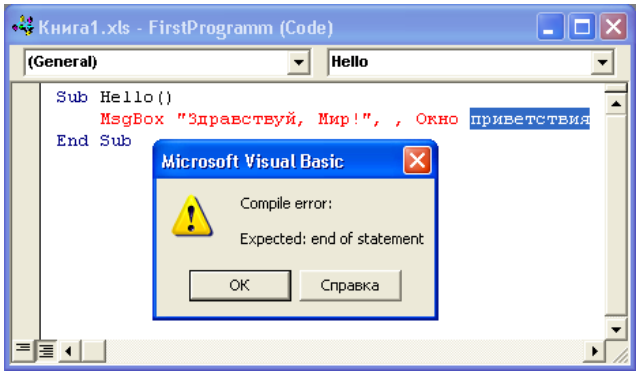
После успешного завершения синтаксического анализа строка подвергается трансляции, т.е. переводу к такому виду, в котором она может быть непосредственно выполнена на компьютере. Если синтаксический анализ и трансляция прошли успешно, VBA раскрашивает различные части строки в

разные цвета в соответствии с правилами, которые были изложены выше. Если же в строке будет обнаружена ошибка, то VBA локализует ее (на сколько это ему удается) и выводит соответствующее сообщение.

Например, если в последней приведенной выше процедуре в инструкции обращения к функции ***MsgBox*** будут пропущены двойные кавычки вокруг третьего аргумента, то VBA интерпретирует слово `Окно` не как часть строки, подлежащей выводу в заголовке окна сообщения, а как имя очередного аргумента.

```
MsgBox "Здравствуй, Мир!", , Окно приветствия
```

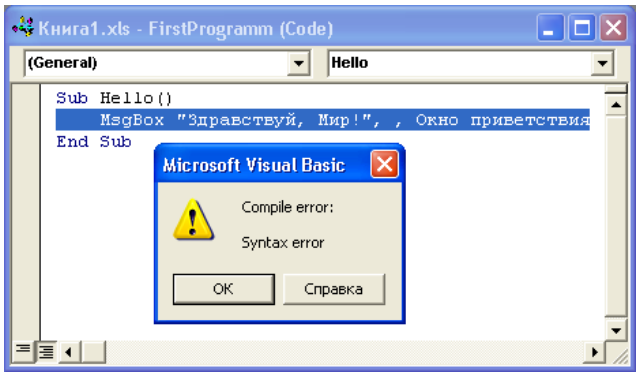
Поскольку слово `Окно` было воспринято как имя аргумента, то согласно синтаксису обращения к функции, после него должна идти или запятая, или оно должно быть последним в списке. Но вместо этого VBA обнаруживает пробел и еще одно слово. Поэтому он выделяет первое «подозрительное» слово, фиксирует синтаксическую ошибку и выводит окно сообщения:



Согласно этому сообщению VBA обнаружил ошибку трансляции (**Compile error**) – **Ожидается: конец инструкции** (**Expected: end of statement**).

Если при этом щелкнуть кнопку **Справка** – будет выведена справочная информация по данной ошибке, если же **OK** – возврат в окно документа **Code**. После этого необходимо исправить обнаруженную VBA ошибку – в данном случае заключить третий аргумент функции ***MsgBox*** в двойные кавычки. Если

же ошибку не исправлять, а просто переместить курсор в другую строку, VBA фиксировать ее больше не будет – *синтаксический анализ и трансляция строки выполняется, только если курсор покидает ее после* ***редактирования***. При этом если запустить на выполнение процедуру, которая содержит синтаксическую ошибку, будет выделена вся строка, и выведено соответствующее окно сообщения.



В данном случае информация об ошибке менее конкретна – локализована только строка, содержащая ошибку, и сообщается, что ошибка синтаксическая.

VBA обнаруживает достаточно много ошибок в коде программы. Однако не все из них он может определить достаточно точно. В таких ситуациях все, что может сделать VBA – это выдать сообщения о том, что в тексте программы имеется ошибка. Дополнительную информацию при этом можно получить, если поместить курсор на интересующее слово и нажать клавишу **F1**.

Вполне возможно написать такую программу, которая не будет содержать синтаксических ошибок, но при этом правильно ее выполнить на компьютере будет не возможно. Ошибки такого рода называются *семантическими ошибками*, или *ошибками времени выполнения* (*run-time error*). Подобные ошибки могут быть вызваны различными причинами:

- ❒ пропущенными обязательными аргументами процедуры или функции,
- ❒ несоответствием типов аргументов,
- ❒ попыткой доступа к несуществующему ресурсу, либо
- ❒ просто ошибкой в логике работы программы.

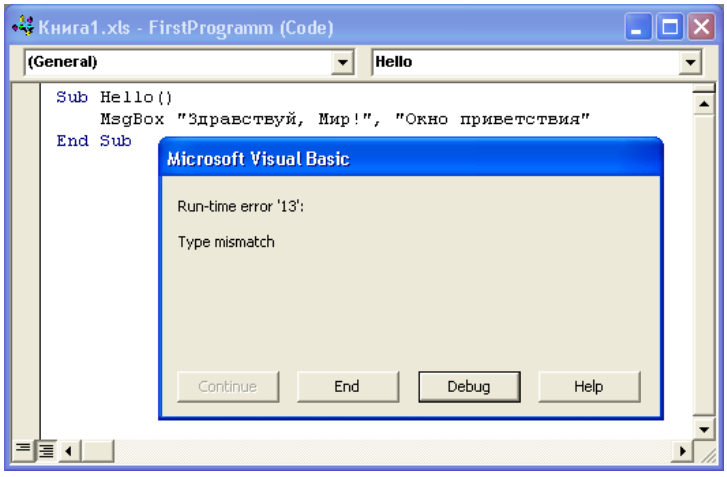
Например, если в инструкции обращения к функции ***MsgBox*** будет «потеряна» одна из двух подряд идущих запятых:

```
MsgBox "Здравствуй, Мир!", "Окно приветствия"
```

В этом случае VBA никаких синтаксических ошибок не обнаружит, поскольку все правила обращения к функции ***MsgBox*** соблюдены:

- текстовые данные заключены в двойные кавычки,
- аргументы разделены запятыми,
- все аргументы в ней, кроме первого, необязательные, поэтому наличие только двух из них вполне допустимо.

Однако, того, что текстовая строка `"Окно приветствия"` должна быть третьим, а не вторым параметром, на этапе синтаксического анализа и трансляции VBA обнаружить не в состоянии. Такая ошибка будет обнаружена лишь на этапе выполнения, и то лишь потому, что второй параметр функции ***MsgBox*** должен быть числом, а не строкой символов. По этому поводу будет выдано необходимое сообщение об ошибке:



Оно гласит о том, что произошла ошибка времени выполнения 13 (**Run-time error '13'**) – недопустимое смешивание (или несоответствие) типов (**Type mismatch**)

В окне сообщения об ошибке имеется несколько кнопок, использовать которые можно следующим образом:

- **Continue** (**Продолжить**). Некоторые ошибки допускают продолжение выполнения приостановленной в связи с этим процедуры, и щелчок по этой кнопке вызывает продолжение выполнения приостановленной процедуры. Если же кнопка недоступна, то продолжение выполнения процедуры, содержащей ошибку, не возможно.
- **End** (**Завершить**). Щелчок по этой кнопке вызывает завершение процедуры.
- **Debug** (**Отладка**). Щелчок по этой кнопке вызывает перевод VBA в режим отладки. При этом можно попытаться также и устранить ошибку.
- **Help** (**Помощь**). Щелчок по этой кнопке вызывает справочную систему VBA с описанием возникшей ошибки

12. Печать

Печать текста программы на бумажный носитель, т.е. получение твердой копии (hard copy) выполняется с целью ее архивирования, более подробного изучения и т.д. Кстати, изучать логику работы программы гораздо удобнее, если она отпечатана на бумаге.

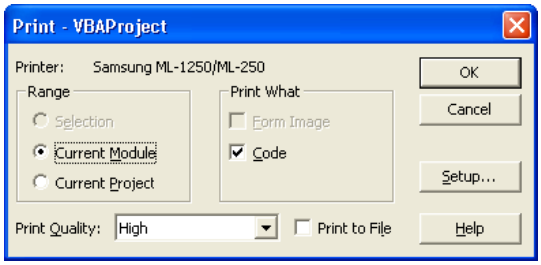
Вывести на печать можно:

- все модули текущего проекта, или
- только выделенный модуль, или
- только выделенный фрагмент модуля.

В редакторе Visual Basic печать выполняется по следующему алгоритму:

1. Выделить подлежащий печати в окне **Project** проект или модуль, или в окне **Code** – фрагмент кода.
2. Активизировать окно диалога настройки параметров печати **Print**:
   - выполнив команду **File ⇨ Print**, или
   - нажав сочетание клавиш **Ctrl** + **P**.

3. В раскрывшемся окне диалога **Print**:



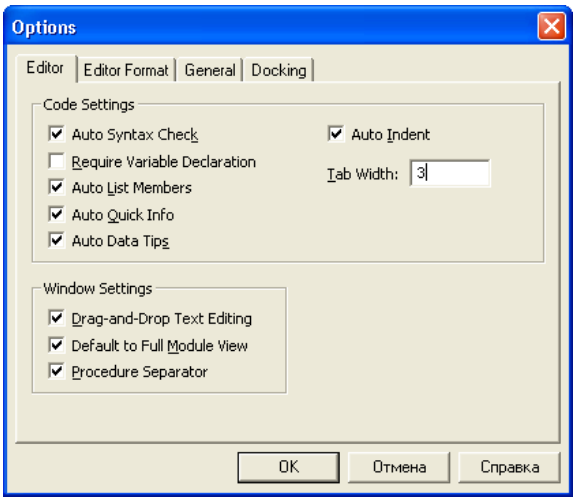
- ❒ В группе **Range** (**Диапазон**) указать диапазон печати:
  - **Selection** – выделенный фрагмент текста,
  - **Current module** – текущий модуль,
  - **Current Project** – текущий проект.
- ❒ В раскрывающемся списке **Print Quality** (**Качество печать**) выставить качество печати – от высокого (**High**) до чернового (**Draft**).
- ❒ Установить флажок **Print to File** (**Печать в файл**), чтобы осуществить вывод в файл, или сбросить его – чтобы направить вывод на принтер.
- ❒ Нажать кнопку **Setup** (**Установка**), чтобы установить дополнительные, зависящие от имеющегося принтера, параметры.
- ❒ Щелкнуть по кнопке **OK**, чтобы начать вывод.

Печать в редакторе Visual Basic имеет некоторые особенности, по сравнению с печатью документов в других текстовых редакторах – в нем, в частности, отсутствует средство предварительного просмотра выводимого на печать документа.

13. Настройка редактора Visual Basic

Приведенные выше приемы работы с редактором Visual Basic сильно зависят от его настроек. Изменить их можно с помощью окна диалога **Options**, которое активизируется по команде **Tools ⇨ Options** (**Сервис ⇨ Параметры**). На вкладке **Editor** (**Редактор**) расположены элементы

управления, которые устанавливают основные параметры работы редактора Visual Basic связанные с редактированием кода, следующим образом:



1. в группе **Code Settings** (**Кодовые установки**) устанавливаются параметры работы с текстом в окне **Code**:
   - ❒ флажок **Auto Syntax Check** (**Автоматический контроль синтаксиса**) определяет, должен ли Visual Basic автоматически проверять правильность синтаксиса после того, как курсор покинет строку кода.
   - ❒ флажок **Require Variable Declaration** (**Требовать объявления переменных**) определяет, требуются ли явные декларации переменных в модулях. Установка этого флажка добавляет инструкцию ***Option Explicit*** к общим декларациям в каждом новом модуле.
   - ❒ флажок **Auto List Member** (**Автоматический список элементов**) определяет, отображать ли список допустимых значений редактируемого компонента программы.
   - ❒ флажок **Auto Quick Info** (**Автоматическая быстрая информация**) определяет, отображать ли вспомогательное окно сведений с информацией о прототипе редактируемой функции, и ее параметрах.

- ❒ флажок **Auto Data Tips** (**Автоматические подсказки по данным**) определяет, отображать ли значение переменной, когда курсор указывает на нее. Используется только в режиме отладки (**Прерывание**, **Приостанов**) **Break**.
- ❒ флажок **Auto Indent** (**Автоматический отступ**) определяет, устанавливать ли выравнивание последующих строк по отступу предыдущей строки.
- ❒ поле редактирования **Tab Width** (**Ширина табуляции**) определяет ширину отступов табуляции, которая может быть в пределах 1÷32 знака.

2. в группе **Window Settings** (**Установки окна**) устанавливаются параметры окна:
   - ❒ флажок **Drag-and-Drop Text Editing** (**Редактирование текста методом Перетащить и Отпустить**) определяет, можно ли перемещать текст между окнами методом Перетащить и Отпустить.
   - ❒ флажок **Default to Full Module View** (**Установить по умолчание полное отображение модуля**) определяет режим отображения для вновь открываемых в окне **Code** модулей – либо всего содержимого модуля, либо только отдельных процедур. Он не влияет на уже отображаемые модули.
   - ❒ флажок **Procedure Separator** (**Разделитель процедур**) позволяет отображать, или прятать, разделительную линию в конце каждой процедуры модуля.

Список источников

1. Информатика. Базовый курс / Под ред. С. В. Симоновича – СПб: Издательство «Питер», 2000. – 640 с: ил
2. Зубов Ф. Н. Microsoft Windows 2000 / Планирование, развертывание, установка. – 2-ое изд. испр. – М.: Издательско-торговый дом «Русская Редакция», 2000. – 592 с.: ил.
3. Андреев А. Г. и др. Microsoft Windows 2000 Professional . Русская вер-сия / Под общ. ред. А. Н. Чекмарева и Д. Б. Вишнякова. – СПб.: БХВ-Петербург, 2001. – 752 с.: ил.
4. Microsoft Word 2000. Шаг за шагом: Практ. Пособие. / Пер. с англ. – М.: Издательство ЭКОМ, 1999. – 464 с.; ил.
5. Рабич Ч. Эффективная работа с Microsoft Word 2000 – СПб.: Издательство «Питер», 2000. – 944 с.: ил.
6. Беленький Ю. М., Власенко С. Ю. Microsoft Word 2000. Наиболее полное руководство. – СПб.: БХВ – Санкт-Петербург, 1999. – 992 с.
7. А. Горячев, Ю. Шафрин. Практикум по информационным технологиям. – М.: Лаборатория базовых знаний, 1999. – 272 с.
8. Шафрин Ю. А. Информационные технологии: В 2 ч. – М.: Лабор. Базовых Знаний, 1999.
9. Гарнаев А. Ю. Excel, VBA, Internet в экономике и финансах. – СПб.: БХВ-Петербург, 2001. – 816 с.: ил.
10. Гарнаев А. Ю. Использование MS Excel и VBA в экономике и финансах. – СПб.: БХВ-Санкт-Петербург, 1999. – 336 с.: ил.
11. Долженков В. А., Колеников Ю. В. Самоучитель Microsoft Excel 2000. – СПб.: БХВ – Петербург, 2002.– 368 с.: ил.
12. Брукшир, Дж., Гленн. Введение в компьютерные науки. – М.: Издательский дом «Вильямс», 2001.-688 с.

НАВЧАЛЬНЕ ВИДАННЯ

ПОГРЕБНЯК Борис Іванович

КОМП'ЮТЕРНА ТЕХНІКА ТА ПРОГРАМУВАННЯ

КОНСПЕКТ ЛЕКЦІЙ

(для студентів 1-го курсу заочної форми навчання освітньо-кваліфікаційного рівня бакалавр, напряму підготовки 6.070101 «Транспортні технології» (за видами транспорту))

(Рос. мовою)

Відповідальний за випуск *М. І. Самойленко*

За авторською редакцією

Комп'ютерне верстання *О. А. Балашова*

План 2011, поз. 191Л

Підп. до друку 18.11.2011 р. Формат 60×84/16

Друк на ризографі. Ум. друк. арк. 7,9

Тираж 50 пр. Зам. №

Видавець і виготовлювач:
Харківська національна академія міського господарства,
вул. Революції, 12, Харків, 61002
Електронна адреса: rectorat@ksame.kharkov.ua
Свідоцтво суб'єкта видавничої справи:
ДК № 4064 від 12.05.2011 р.